

handbook of statistics 31

Machine Learning:
Theory and Applications

Edited by
V. Govindaraju
C.R. Rao



HANDBOOK OF STATISTICS
VOLUME 31

Handbook of Statistics

VOLUME 31

General Editor

C.R. Rao

C.R. Rao AIMSCS, University of Hyderabad Campus, Hyderabad, India



ELSEVIER

Amsterdam • Boston • Heidelberg • London • New York • Oxford
Paris • San Diego • San Francisco • Singapore • Sydney • Tokyo

Volume 31

Machine Learning: Theory and Applications

Edited by

Venu Govindaraju

University at Buffalo, Buffalo, USA

C.R. Rao

C.R. Rao AIMSCS, University of Hyderabad Campus, Hyderabad, India



Amsterdam • Boston • Heidelberg • London • New York • Oxford
Paris • San Diego • San Francisco • Singapore • Sydney • Tokyo

North-Holland is an imprint of Elsevier



North-Holland is an imprint of Elsevier
The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK
Radarweg 29, PO Box 211, 1000 AE Amsterdam, The Netherlands

First edition 2013

Copyright © 2013 Elsevier B.V. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means electronic, mechanical, photocopying, recording or otherwise without the prior written permission of the publisher.

Permissions may be sought directly from Elsevier's Science & Technology Rights Department in Oxford, UK: phone (+44) (0) 1865 843830; fax (+44) (0) 1865 853333; email: permissions@elsevier.com. Alternatively you can submit your request online by visiting the Elsevier web site at <http://elsevier.com/locate/permissions>, and selecting *Obtaining permission to use Elsevier material*.

Notice

No responsibility is assumed by the publisher for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions or ideas contained in the material herein. Because of rapid advances in the medical sciences, in particular, independent verification of diagnoses and drug dosages should be made.

British Library Cataloguing in Publication Data

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

Library of Congress Cataloging-in-Publication Data

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

ISBN: 978-0-444-53859-8

ISSN: 0169-7161

For information on all North-Holland publications
visit our web site at store.elsevier.com

Printed and bound in Great Britain
13 14 15 16 10 9 8 7 6 5 4 3 2 1



Working together
to grow libraries in
developing countries

www.elsevier.com • www.bookaid.org

Table of Contents

Volume 31 Handbook of Statistics

Contributors: Vol. 31 xi

Preface to Handbook Volume – 31 xv

Introduction xvii

Part I: Theoretical Aspects 1

Ch. 1. The Sequential Bootstrap 3
P.K. Pathak and C.R. Rao

- 1. Introduction 4
- 2. A sequential bootstrap resampling scheme 7
- 3. Bootstrapping empirical measures with a random sample size 9
- 4. Convergence rates for the sequential bootstrap 12
- 5. Second-order correctness of the sequential bootstrap 14
- 6. Concluding remarks 17
 - Acknowledgments 17
 - References 18

Ch. 2. The Cross-Entropy Method for Estimation 19
Dirk P. Kroese, Reuven Y. Rubinstein, and Peter W. Glynn

- 1. Introduction 19
- 2. Estimation setting 20
- 3. Extensions 26
 - Acknowledgment 33
 - References 33

Ch. 3. The Cross-Entropy Method for Optimization 35
*Zdravko I. Botev, Dirk P. Kroese, Reuven Y. Rubinstein,
and Pierre L'Ecuyer*

- 1. Introduction 35
- 2. From estimation to optimization 36
- 3. Applications to combinatorial optimization 40
- 4. Continuous optimization 50
- 5. Summary 57
- References 57

Ch. 4. Probability Collectives in Optimization 61
David H. Wolpert, Stefan R. Bieniawski, and Dev G. Rajnarayan

- 1. Introduction 61
- 2. Delayed sampling theory 64
- 3. Delayed sampling experiments 70
- 4. Immediate sampling theory 82
- 5. Immediate sampling experiments 85
- 6. Conclusion 97
- References 98

Ch. 5. Bagging, Boosting, and Random Forests Using R 101
*Hansen Bannerman-Thompson, M. Bhaskara Rao,
and Subramanyam Kasala*

- 1. Introduction 101
- 2. Data sets and rationale 103
- 3. Bagging 107
- 4. Boosting 113
- 5. Do Bagging and Boosting really work? 115
- 6. What is a classification tree? 116
- 7. Classification tree versus logistic regression 126
- 8. Random forest 127
- 9. Random forest, genetics, and cross-validation 134
- 10. Regression trees 139
- 11. Boosting using the R package, ada 144
- 12. Epilog 149
- References 149

Ch. 6. Matching Score Fusion Methods 151
Sergey Tulyakov and Venu Govindaraju

- 1. Introduction 151
- 2. Matching systems 156
- 3. Selected approaches to fusion in matching systems 158
- 4. Operating modes of matching systems 162
- 5. Complexity types of classifier combination methods 163
- 6. Modeling matching score dependencies 165
- 7. Score combination applications 167
- 8. Conclusion 169
- Appendices 169

A. Proof of Claim 4	169
B. Proof of Claim 5	171
References	173

Part II: Object Recognition 177

Ch. 7. Statistical Methods on Special Manifolds for Image and Video Understanding 179 *Pavan Turaga, Rama Chellappa, and Anuj Srivastava*

1. Introduction	179
2. Some motivating examples	180
3. Differential geometric tools	181
4. Common manifolds arising in image analysis	185
5. Applications in image analysis	187
6. Summary and discussion	198
Acknowledgments	198
References	198

Ch. 8. Dictionary-Based Methods for Object Recognition 203 *Vishal M. Patel and Rama Chellappa*

1. Introduction	203
2. Sparse representation	204
3. Dictionary learning	210
4. Concluding remarks	222
References	222

Ch. 9. Conditional Random Fields for Scene Labeling 227 *Ifeoma Nwogu and Venu Govindaraju*

1. Introduction	227
2. Overview of CRF	229
3. Scene parsing	236
4. More recent implementations of CRF scene labelings	242
5. Conclusion and future directions	245
References	245

Ch. 10. Shape-Based Image Classification and Retrieval 249 *N. Mohanty, A. Lee-St. John, R. Manmatha, and T.M. Rath*

1. Introduction	250
2. Prior work	251
3. Classification and retrieval models	252
4. Features	257
5. Classification experiments	261
6. Retrieval	262

7.	Multiple class labels	265
8.	Summary and conclusions	266
	References	266
Ch. 11. Visual Search: A Large-Scale Perspective		269
<i>Robinson Piramuthu, Anurag Bhardwaj, Wei Di, and Neel Sundaresan</i>		
1.	Introduction	269
2.	When is big data important?	272
3.	Information extraction and representation	273
4.	Matching images	282
5.	Practical considerations: memory footprint and speed	285
6.	Benchmark data sets	289
7.	Closing remarks	291
	References	293
Part III: Biometric Systems		299
Ch. 12. Video Activity Recognition by Luminance Differential Trajectory and Aligned Projection Distance		301
<i>Haomian Zheng, Zhu Li, Yun Fu, Aggelos K. Katsaggelos, and Jane You</i>		
1.	Introduction	302
2.	Related work	303
3.	Problem formulation	305
4.	DLFT and LAPD solutions	312
5.	Experiments	314
6.	Conclusion	323
	References	324
Ch. 13. Soft Biometrics for Surveillance: An Overview		327
<i>D.A. Reid, S. Samangooei, C. Chen, M.S. Nixon, and A. Ross</i>		
1.	Introduction	327
2.	Performance metrics	329
3.	Incorporating soft biometrics in a fusion framework	330
4.	Human identification using soft biometrics	333
5.	Predicting gender from face images	344
6.	Applications	346
7.	Conclusion	349
	References	350
Ch. 14. A User Behavior Monitoring and Profiling Scheme for Masquerade Detection		353
<i>Ashish Garg, Shambhu Upadhyaya, and Kevin Kwiat</i>		
1.	Introduction	354
2.	Related work	356
3.	Support Vector Machines (SVMs)	358

4.	Data collection, feature extraction, and feature vector generation	361
5.	Experimental design	367
6.	Discussion and conclusion	376
	Acknowledgments	376
	References	377
Ch. 15. Application of Bayesian Graphical Models to Iris Recognition		381
<i>B.V.K. Vijaya Kumar, Vishnu Naresh Boddeti, Jonathon M. Smereka, Jason Thornton, and Marios Savvides</i>		
1.	Introduction	381
2.	Gabor wavelet-based matching	383
3.	Correlation filter-based iris matching	386
4.	Bayesian graphical model for iris recognition	390
5.	Summary	397
	Acknowledgments	397
	References	397
Part IV: Document Analysis		399
Ch. 16. Learning Algorithms for Document Layout Analysis		401
<i>Simone Marinai</i>		
1.	Introduction	401
2.	Pixel classification	405
3.	Zone classification	406
4.	Connected component classification	409
5.	Text region segmentation	411
6.	Region classification	412
7.	Functional labeling	415
8.	Conclusion	416
	References	416
Ch. 17. Hidden Markov Models for Off-Line Cursive Handwriting Recognition		421
<i>Andreas Fischer, Volkmar Frinken, and Horst Bunke</i>		
1.	Introduction	421
2.	Serialization of handwriting images	423
3.	HMM-based text line recognition	426
4.	Outlook and conclusions	438
	Acknowledgment	439
	References	439
Ch. 18. Machine Learning in Handwritten Arabic Text Recognition		443
<i>Utkarsh Porwal, Zhixin Shi, and Srirangaraj Setlur</i>		
1.	Introduction	443
2.	Arabic script—challenges for recognition	444
3.	Learning paradigms	447
4.	Features for text recognition	455

5. Models for recognition	460
6. Conclusion	467
References	468

Ch. 19. Manifold Learning for the Shape-Based Recognition of Historical Arabic Documents 471
Mohamed Cheriet, Reza Farrahi Moghaddam, Ehsan Arabnejad, and Guoqiang Zhong

1. Introduction	471
2. Problem statement	475
3. Manifold learning	475
4. Feature extraction	480
5. Experimental results	481
6. Conclusion and future prospects	488
Acknowledgments	489
References	489

Ch. 20. Query Suggestion with Large Scale Data 493
Nish Parikh, Gyanit Singh, and Neel Sundaresan

1. Introduction	493
2. Terminology	499
3. Approaches to generation of Query Suggestions	501
4. Evaluation methods of QS	508
5. Properties of large scale data	509
6. Query Suggestion in practice	513
7. Closing remarks	515
References	516

Subject Index 519

Contributors: Vol. 31

- A. Lee-St. John, *Department of Computer Science, Mt. Holyoke College, S. Hadley, MA 01075, USA* (Ch. 10).
- A. Ross, *Lane Department of Computer Science and Electrical Engineering, West Virginia University, USA* (Ch. 13).
- Aggelos K. Katsaggelos, *Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA* (Ch. 12).
- Andreas Fischer, *University of Bern, Institute of Computer Science and Applied Mathematics, Neubrückstrasse 10, 3012 Bern, Switzerland* (Ch. 17).
- Anuj Srivastava, *Department of Statistics, Florida State University, Tallahassee, FL, USA* (Ch. 7).
- Anurag Bhardwaj, *eBay Research Labs, San Jose, CA, USA* (Ch. 11).
- Ashish Garg, *Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY 14260, USA* (Ch. 14).
- B.V.K. Vijaya Kumar, *Electrical and Computer Engineering, Carnegie, Mellon University, Pittsburgh, PA, USA* (Ch. 15).
- C. Chen, *Lane Department of Computer Science and Electrical Engineering, West Virginia University, USA* (Ch. 13).
- C.R. Rao, *Department of Statistics, Pennsylvania State University, University Park, PA 16802, USA; University at Buffalo, Buffalo, NY 14260, USA* (Ch. 1).
- D.A. Reid, *School of Electronics and Computer Science, University of Southampton, UK* (Ch. 13).
- David H. Wolpert, *Santa Fe Institute, NASA Ames Research Center MailStop 269-1, Moffett Field, CA 94035-1000, USA* (Ch. 4).
- Dev G. Rajnarayan, *Desktop Aeronautics, Inc., 1900 Embarcadero Road, Suite 101, Palo Alto, CA 94303, USA* (Ch. 4).
- Dirk P. Kroese, *School of Mathematics and Physics, The University of Queensland, Brisbane 4072, Australia* (Ch. 2).
- Dirk P. Kroese, *School of Mathematics and Physics, The University of Queensland, Brisbane 4072, Australia* (Ch. 3).
- Ehsan Arabnejad, *Synchromedia Laboratory for Multimedia Communication in Telepresence, École de technologie supérieure, Montreal, QC, Canada H3C 1K3* (Ch. 19).

- Guoqiang Zhong, *Synchromedia Laboratory for Multimedia Communication in Telepresence, École de technologie supérieure, Montreal, QC, Canada H3C 1K3* (Ch. 19).
- Gyanit Singh, *eBay Research Labs, San Jose, CA, USA* (Ch. 20).
- Hansen Bannerman-Thompson, *University of Cincinnati, USA* (Ch. 5).
- Haomian Zheng, *Department of Computing, Hong Kong Polytechnic University, Hong Kong* (Ch. 12).
- Horst Bunke, *University of Bern, Institute of Computer Science and Applied Mathematics, Neubrückstrasse 10, 3012 Bern, Switzerland* (Ch. 17).
- Ifeoma Nwogu, *Department of Computer Science and Engineering, University at Buffalo, SUNY, Buffalo, NY 14260, USA* (Ch. 9).
- Jane You, *Department of Computing, Hong Kong Polytechnic University, Hong Kong* (Ch. 12).
- Jason Thornton, *Lincoln Laboratory, Lexington, MA, USA* (Ch. 15).
- Jonathon M. Smereka, *Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA* (Ch. 15).
- Kevin Kwiat, *Air Force Research Laboratory, 525 Brooks Rd., Rome, NY 13441, USA* (Ch. 14).
- M. Bhaskara Rao, *University of Cincinnati, USA* (Ch. 5).
- M.S. Nixon, *School of Electronics and Computer Science, University of Southampton, UK* (Ch. 13).
- Marios Savvides, *Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA* (Ch. 15).
- Mohamed Cheriet, *Synchromedia Laboratory for Multimedia Communication in Telepresence, École de technologie supérieure, Montreal, QC, Canada H3C 1K3* (Ch. 19).
- N. Mohanty, *Google, Mountain View, CA 94043, USA* (Ch. 10).
- Neel Sundaresan, *eBay Research Labs, San Jose, CA, USA* (Chs. 11, 20).
- Nish Parikh, *eBay Research Labs, San Jose, CA, USA* (Ch. 20).
- P.K. Pathak, *Department of Statistics and Probability, Michigan State University, East Lansing, MI 48824, USA* (Ch. 1).
- Pavan Turaga, *School of Arts, Media, Engineering, School of Electrical, Computer, Energy Engineering, Arizona State University, Tempe, AZ, USA* (Ch. 7).
- Peter W. Glynn, *Department of Management Science and Engineering, Stanford University, CA, USA* (Ch. 2).
- Pierre L'Ecuyer, *Department of Computer Science and Operations Research, Université deMontréal, Montréal, Québec, Canada H3C 3J7* (Ch. 3).
- R. Manmatha, *Department of Computer Science, University of Massachusetts, Amherst, MA 01003, USA* (Ch. 10).
- Rama Chellappa, *Department of Electrical and Computer Engineering and UMIACS, University of Maryland, College Park, MD 20742, USA* (Chs. 7, 8).
- Reuven Y. Rubinstein, *Faculty of Industrial Engineering and Management, Technion, Haifa, Israel* (Chs. 2, 3).
- Reza Farrahi Moghaddam, *Synchromedia Laboratory for Multimedia Communication in Telepresence, École de technologie supérieure, Montreal, QC, Canada H3C 1K3* (Ch. 19).
- Robinson Piramuthu, *eBay Research Labs, San Jose, CA, USA* (Ch. 11).

- S. Samangooei, *School of Electronics and Computer Science, University of Southampton, UK* (Ch. 13).
- Sergey Tulyakov, *University at Buffalo, USA* (Ch. 6).
- Shambhu Upadhyaya, *Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY 14260, USA* (Ch. 14).
- Simone Marinai, *Dipartimento di Sistemi e Informatica, Universit degli Studi di Firenze, Italy* (Ch. 16).
- Srirangaraj Setlur, (Ch. 18).
- Stefan R. Bieniawski, *Boeing Research & Technology, P.O. Box 3707, MC 42-51, Seattle, WA, USA* (Ch. 4).
- Subramanyam Kasala, *University of North Carolina, USA* (Ch. 5).
- T.M. Rath, *Google, Mountain View, CA 94043, USA* (Ch. 10).
- Utkarsh Porwal, (Ch. 18).
- Venu Govindaraju, *Department of Computer Science and Engineering, University at Buffalo, SUNY, Buffalo, NY 14260, USA* (Chs. 6, 9).
- Vishal M. Patel (Ch. 8).
- Vishnu Naresh Boddeti, *Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA* (Ch. 15).
- Volkmar Frinken, *University of Bern, Institute of Computer Science and Applied Mathematics, Neubrückstrasse 10, 3012 Bern, Switzerland* (Ch. 17).
- Wei Di, *eBay Research Labs San Jose, CA, USA* (Ch. 11).
- Yun Fu, *Department of Computer Science and Engineering, State University of New York, Buffalo, NY, USA* (Ch. 12).
- Zdravko I. Botev, *Department of Computer Science and Operations Research, Université de Montréal, Montréal, Québec, Canada H3C 3J7* (Ch. 3).
- Zhixin Shi, (Ch. 18).
- Zhu Li, *Media Lab, FutureWei (Huawei) Technology, Bridgewater, NJ, USA* (Ch. 12).

This page is intentionally left blank

Preface to Handbook Volume – 31

Thought leaders in academia, industry, and government are focused on the explosion of data being generated around us on a continuous basis in almost every field of scientific endeavor. This has led to a fundamental paradigm shift in how scientific progress is achieved. Data analytics has now become central to knowledge discovery in what Jim Gray from Microsoft Research has termed “The Fourth Paradigm.” In the last few decades, the pace of scientific discovery increased sharply with the advent of computers and progressively easier access to computing resources. Science turned computational (“the third paradigm”) with the ability to generate simulated data produced by the modeling of complex phenomena and the development of algorithms that could make use of the computational power for analysis of data.

This new shift in paradigm to data-driven research re-focuses the emphasis away from raw computational power to the development of specialized algorithms for learning from data and reasoning with the data. Thus Machine Learning (ML) techniques have come to play a central role in automatic data processing and analytics across a wide spectrum of research domains. However, the lack of well-defined principles in choosing ML algorithms suitable for a given problem remains a major challenge. Empirical determination of parameters such as the size of training data, the number of distinct labels, interpretable decision boundaries, and choice of a suitable ML algorithm is more of an art than science.

This book reviews a broad spectrum of topics ranging from theory to applications in object recognition, document recognition, and biometrics. It includes a selection of current topics addressed by leading researchers in the field. The primary intended audience for this book is the research community at academia and industrial research labs. The secondary audience is graduate students working on doctoral dissertations. The authors would like to acknowledge the support of the chapter contributors for the timely submissions of their high quality research work and the reviewers for their important suggestions. We are indebted to Dr. Sergey Tulyakov and Amal Harb at the Center for Unified Biometrics and Sensors (CUBS), University at Buffalo, State University of New York, USA, for helping with the organization of files, setting up the web interface for chapter submission and the review process.

Amherst, NY, USA
March 1, 2013

Venu Govindaraju
C. R. Rao

This page is intentionally left blank

Introduction

Modern machine learning methods are utilized in a wide variety of applications and provide an unprecedented level of automation in all areas of life. They have also spurred an increased research interest in the statistical foundations of learning methods and discovering best possible algorithms to handle complex data collected by constantly improving and proliferating sensors. The research areas are becoming more specialized, and it might be difficult for researchers to get a full picture of the diversity of topics and problems investigated in this field.

This book attempts to capture the current progress in multiple research areas related to machine learning. It contains a collection of chapters written by different researchers in machine learning field, looking at this field from different points of view. Both theoretical and practical approaches are represented here. We organized the material in four parts. The first part contains chapters devoted to the theoretical aspects of machine learning. Second part has chapters dealing with general machine learning applications, e.g., the recognition and modeling of arbitrary objects. Part three describes the applications of learning methods in the biometric applications, and, finally, the fourth part gives examples of employing learning methods in document analysis systems. Below we provide a more detailed description of book's chapters.

Part I: Theoretical aspects

The first part of the book contains six chapters on some theoretical issues arising in machine learning problems. Typically, many machine learning methods are being developed to address a specific learning task, and are valued by their practical performance and not by theoretical soundness. The task of many theory-oriented researchers is to explain why some algorithm has good performance, and what possible steps could be taken to improve its performance. The results might be applicable to many learning algorithms, but due to more difficult and time-consuming nature, theoretical research is not as widely covered as practical machine learning methods.

The estimation of the model's parameters from the training data is a typical problem encountered in learning procedures. Instead of a single estimate of a parameter from the given set of training samples, the bootstrap methods produce multiple estimates from randomly chosen subsets of original training data and

derive a more precise estimate of the parameter, as well as, the confidence intervals for these estimates. The first chapter of our book explores a particular method of generating random training data subsets—*sequential bootstrap*. Whereas simple bootstraps with resampling can produce a variable number of repeated samples in the generated subsets, the subsets generated by sequential bootstrap technique have same proportion of repeated samples. Authors present some properties of sequential bootstrap, and in particular prove its superior performance relative to simple bootstrap methods.

Next two chapters discuss the *cross-entropy* method of statistical sampling. The idea of this method is that instead of using original distribution f of the random variable we might attempt to use a distribution g more suitable for the sampling objective. For example, if we want to generate some rare events according to f , we find g so that it is more concentrated in the region of rare events of f . The cross-entropy method performs an iterative procedure for approximating the optimal helper distribution g^* ; the cross-entropy measure indicates how the n th iteration of g is close to g^* . The method is applied to estimation and optimization tasks in corresponding chapters.

Chapter 4 discusses the optimization problem from a different perspective. It treats optimized system as a blackbox and assumes that the estimation of the performance for specific vector of parameters takes long time, e.g., as in genetic algorithms or simulated annealing. Instead of applying some heuristic function for choosing the next vector of parameters for system evaluation, the *Probability Collectives* approach explicitly analyzes the distribution of such vectors, and samples next vector of parameters from it. Two techniques of sampling Probability Collectives, delayed and immediate, are presented in the chapter and extensive examples of using these techniques are given.

The next chapter has a more practical orientation—it shows how different machine learning techniques can be implemented and tested with the help of open source software R. It pays particular attention to the tree-based classification and regression methods, as well as, the construction of classifier systems by bagging and boosting. It serves as a good review and practical recipe for using the decision trees and random forests learning methods.

Chapter 6 addresses the problem of combining the results produced by multiple matching systems. Authors define a matching system as a special type of classifier, which derives the confidence scores for each class separately from other classes. The chapter investigates the different ways on how the combination functions for such systems can be constructed and learned from the training data. The achieved results are summarized in five claims, and the proofs for the claims, either theoretical or experimental, are provided.

Part II: Object recognition

The earlier applications of machine learning methods dealt mostly with narrowly defined tasks, such as classification of objects in two or a very small number of classes. The recent research interest has shifted more toward universal learning methods which is able to recognize a large number of classes or properties of objects. Such

research is made possible by the increased computing power of modern computers and the increased number of different sensors, and the desire to utilize all available resources. In this part of the book we collected five chapters describing such general learning methods, mostly for processing visual data.

One idea, which permeates the current research area of object recognition, is the low dimensional nature of object's observations and the possibility to create the models which utilize it. One example of such model is the principal component analysis (PCA), which assumes that the objects occupy a hyperplane in higher dimensional feature space, and the position of the hyperplane can be approximated from the set of training data. Given the success achieved by the recognition methods utilizing such models, the researchers are increasingly more interested in constructing more precise and possibly more complex, e.g., nonlinear, representational models for different object recognition tasks. In Chapter 7 authors assume that the additional constraints on object representation could be imposed due to variability in data collection. For example, if the recognized object is rotated and translated in three-dimensional space, then the set of points describing this object will not change arbitrarily, but will follow a curve in some manifold containing possible object transformations. If we want to compare two objects or calculate the distance between them, we can use the distance imposed by the manifold instead of traditional Euclidean distance. Chapter 7 gives a review of the recognition methods utilizing manifolds, such as special orthogonal group, Stiefel and Grassmann manifolds, symmetric positive definite matrices. The considered applications of manifold representations include curve-based shape recognition, human activity analysis, facial age and expression analysis, 3D object tracking and recognition.

Chapter 8 addresses another kind of object modeling technique—sparse representation coding. This technique assumes that an object could be represented as a linear combination of a small number of basis or dictionary vectors. The basis vectors could be simply given as a set of training feature vectors, or could be learned in order to achieve a better representation power. The main question investigated in this chapter is how the dictionary vectors can be learned even for nonlabeled training dataset. Authors provide a comprehensive review of the methods utilizing such learning and describe some extensions of this technique, such as discriminative learning, nonlinear kernel-based learning, dimensionality reduction coupled with dictionary learning.

The object recognition task is closely related to the problem of object segmentation from the images. Chapter 9 presents an overview of the scene segmentation methods utilizing Conditional Random Fields (CRF). CRFs and other graphical models have been a recent addition to the machine learning methods. The idea of these methods is to not only represent the appearance of the objects or some part of the objects (graph vertices), but also the relationships between different objects or their parts (graph edges). The probabilities for graph vertices and edges can be learned from the training data, and the scene recognition can be performed by fitting learned graphical models. After the introduction of CRFs and discussing the various issues of their training and inference, the authors present specific details on the features suitable for scene recognition—color, geometry, and texture, and provide an example for constructing CRF for scene segmentation. At the end, the examples of more recent CRF used in the literature are given.

The border of the segmented object can be represented by a closed chain of connected image pixels—a contour. The contour representation of the objects was historically widely used in optical character recognition applications, for which the binarization of the image into foreground (characters) and background (paper) was natural and the contours were easily extracted from binarized images. But contours could also be successfully used for general object representation and recognition, and the next chapter presents a series of experiments comparing some of the methods performing such recognition. In particular, the centroid distance and profile features along with different classification algorithms are tested, and it was found out that the profile features coupled with Cross-Media Relevance Model (CMRM) recognizer had best performance. Additional experiments were also performed to investigate the effects of object rotations and class variations on the training and recognition performance.

The final chapter of this part looks at the object recognition task from the perspective of search engine. Given the enormous amount of visual data available on the Internet, the automatic recognition and fast retrieval of relevant images become a necessity. Authors review the variety of the algorithms for the extraction of image features, best template structure, availability of indexing or fast search techniques, query structure, and so on. In particular, the suitability of the following methods is analyzed: use of global and local features, color histograms, few textural features, filter transformation features (wavelets, LBP, SIFT, SURF, etc.), multiple methods for calculating distances between feature vectors, and multiple hashing methods. The authors do not provide ready solutions, but rather stress the difficulty of the problem and pose some additional questions for future research.

Part III: Biometric systems

Biometric person identification and recognizing the natural human interactions as a part of building human–computer interface gained significant amount of interest lately. Different sensors have been produced for capturing multiple biometric modalities (face, fingerprint, iris, etc.) and the abundance of data greatly facilitated the development of automatic methods for their matching in biometric systems, or recognition and interpretation by HCI systems. Four chapters in this area collected in this part of the book certainly do not cover all aspects of biometric research, but can provide an insight on how biometric systems are constructed and operated, as well as, the current state of the art in the processing of human-based sensor input.

The first chapter of this part, Chapter 12, addresses the problem of human body movement and hand gesture recognition. Instead of building some complex action models, e.g., based on HMM, the authors chose template-based matching method. In this method, the samples of each action class are stored as templates, and the classification of a test sample is performed by finding the closest template. The main question for this method is the choice of best template representation and the calculation of the distance between two templates. Authors used PCA for extracting features for each frame and performed experiments testing multiple distance calculation methods. Whereas first method, Differential Luminance Field Trajectory, relied on the set of differences between consecutive frames of the video and

captured temporal information, the second method, Luminance Aligned Projection Distance, performed direct distance calculation between two sequences of feature vectors. Both methods seem to outperform the previously proposed graphical model-based approach, as well as, the distance calculation by traditional DTW method. The combination of both methods is reported to have even better performance.

The traditional biometric modalities, such as fingerprint and face, appear to be sufficient for unique person identification, if not by computer algorithms yet, but possibly by forensic examiners. In contrast to such modalities, some biometric measures might not have such discriminative power. The examples include the height and weight of the person, colors of the skin, hair, and eyes, measurements of body parts, gender, age, race, etc. Such measures are called *soft biometric* modalities. Unlike hard biometric modalities typically requiring dedicated sensors to obtain measurements, many soft biometric modalities can be obtained using remote video camera. The research into automatic person identification using soft biometrics is relatively new, and Chapter 13 provides an extensive review of the state-of-the-art techniques in this area. Of particular interest to the authors are the questions on the discriminative power of different soft biometric measures and on their dependence. Some experimental results are presented for these questions.

The next chapter provides an example of biometric application utilizing the patterns of computer usage by a person. In this application, the user actions, such as mouse movements, key strokes, and mouse clicks, are captured and subsequently used to verify the identity of the currently logged-in user. The application follows a standard methodology for pattern classification algorithms: first, a set of features is extracted from raw recorded data, second, a pattern classifier is trained on training set of feature vectors, and, lastly, the pattern classifier is applied on new data. In particular, the frequencies of specific mouse movements and clicks over some time period and keystroke times are used as features in the described work, and Support Vector Machine classifiers are trained to distinguish each user from all other users. Since the extracted features are not sufficient to uniquely identify each user, current biometric application can be considered as the one using soft biometric modality.

In contrast, to previous chapter, Chapter 15 investigates the use of hard or strong biometric modality—iris. At first, it reviews the traditional algorithm for iris matching of Daugman based on Gabor wavelet feature extraction and Hamming distance calculation for two binary templates, as well as the previously proposed algorithm by authors based on correlation filters. Although these algorithms have a fairly good performance, authors point out that they do not take into consideration possible nonlinear deformations of the irises and demonstrate the example of such deformation in iris images. To solve this problem, authors propose an original algorithm which considers the correlations between local patches of two iris images, and construct a transformation map providing correspondences between patches. By incorporating the likelihood of transformation with the correlation scores, the algorithm accounts for nonlinear deformations and achieves superior performance.

Part IV: Document analysis

The last part of this book is devoted to the applications of machine learning methods to the document analysis problems. Historically, optical character recognition

problem was used as one of the first test beds for the nascent field of machine learning in 1970s and 1980s. To this day, many machine learning techniques get applications in document analysis field, even though they might be first developed for other fields. For instance, Hidden Markov Models were initially used for speech recognition, but were later adapted and became a mainstream technique for handwritten word recognition. Many chapters of this part show similar examples of machine learning methods developed for different areas being used in document analysis field.

The first chapter of this part, Chapter 16, presents a review of methods in document layout analysis. Note that most document analysis system employ a top-down design consisting in consecutive segmentation of text regions, lines, words, and, finally, recognizing words and possibly separate characters. Thus, the layout analysis, i.e., recognition of text foreground and different types of text regions, constitutes an important first part of such systems. This chapter discusses a significant number of previously proposed methods for layout analysis, which are organized with respect to the intended level of segmentation: pixel, zone, connected component, text, text region, general regions, and text regions of particular type. The variety of discussed methods includes the older style classification approaches, e.g., using neural networks to classify foreground and background pixels, as well as more modern methods, e.g., using graphical model approaches (Markov Random fields, Conditional Random Fields) to represent the structure of the document.

Chapter 17 addresses the technique of offline text recognition by means of Hidden Markov Models (HMMs). HMMs are well suited for modeling the sequential data, which is generated by the finite number of possible states in a process. In handwriting recognition, the states usually correspond to pen strokes or other distinct pen movements, and the modeled sequences of states could correspond to characters or whole words. In this chapter, authors review how HMMs for handwriting recognition could be constructed, what image preprocessing should be done, which HMM architectures should be used, how to perform training, and how the trained HMMs can be used for text recognition. The special attention is paid to the questions of incorporating language models into HMM-based text recognition and deriving the final confidence score of recognized text.

The recognition of documents consisting of non-Latin scripts garnered much research interest recently, and, as an example, next two chapters seek to apply multiple machine learning methods to the problem of recognizing Arabic texts. Arabic scripts pose unique challenges to recognition such as great variability in character shapes and difficulty in separating characters from each other. In Chapter 18, authors explore the applications of some novel machine learning methods, e.g., co-training, Deep Belief Networks (DBNs), and Restricted Boltzmann Machines (RBMs), in addition to more traditional neural networks and HMMs-based Arabic word recognition and general questions of feature extraction and character recognition. Chapter 19 concentrates on the subspace modeling of Arabic subwords by PCA, generalized discriminant analysis (GDA)—a variant of linear discriminant analysis performed in a kernel space, locally linear embedding (LLE), and supervised LLE. Most of these techniques have been previously developed for other machine learning tasks, notably, face recognition, and their application for handwriting recognition confirms their usefulness.

The last chapter of the book, Chapter 20, describes a problem of improving user's search engine queries with the help of query suggestions. Instead of looking at one particular document (or web page), it investigates the relationships between user's query and the sets of documents on particular topics, as well as, the relationships between different queries. The authors clearly define the specific terminology of this research field and list a number of possible query suggestion tasks. The chapter also provides an extensive review of the training methods with the particular stress on the distance measures between queries and documents, methods for collecting training data, e.g., by keeping logs of consecutive queries by the same user, methods for evaluating system performance, e.g., measures of coverage and click through rate, the distributions of the results. Finally, some practical suggestions on query suggestion implementations, e.g., dealing with large data sets and presentation of results, are made.

This page is intentionally left blank

Part I: Theoretical Aspects

This page is intentionally left blank

The Sequential Bootstrap

Dedicated to the memory of Kesar Singh (1955–2012),
pioneer of the mathematical foundations of bootstrap
resampling.

P.K. Pathak¹ and C.R. Rao^{2,3}

¹*Department of Statistics and Probability, Michigan State University,
East Lansing, MI 48824, USA*

²*Department of Statistics, Pennsylvania State University, University Park,
PA 16802, USA*

³*University at Buffalo, Buffalo, NY 14260, USA*

Abstract

The main focus of this article is to carefully examine the information content of resampling techniques in bootstrap from a survey sampling point of view. Given an observed sample of size n , resampling for bootstrap involves n repeated trials of simple random sampling with replacement (SRSWR). It is well known that SRSWR does not result in samples that are equally informative (Pathak, 1964). In 1997, Rao et al. introduced a sequential bootstrap resampling scheme in the literature, stemming from the observation made by Efron (1983) that the usual bootstrap samples are supported on average on approximately $0.632n$ of the original data points. The primary goal of our sequential bootstrap was to stabilize the information content of bootstrapped samples as well as to investigate whether consistency and second-order correctness of Efron's simple bootstrap carried over to this sequential case. In Rao et al. (1997), we showed that the main empirical characteristics of the sequential bootstrap asymptotically differ from those of the corresponding simple bootstrap by a magnitude of $O(n^{-3/4})$. In Babu et al. (1999), we established the second-order correctness of the sequential bootstrap for a general class of estimators under regularity conditions analogous to those for the simple bootstrap. In a separate paper, Shoemaker and Pathak (2001) carried out simulations similar to those of Mammen (1992), confirming the consistency of the sequential bootstrap.

More recently, Jiménez-Gamero et al. (2006), Pino-Mejías et al. (2010), Pauly (2011), and others have investigated the sequential bootstrap and its variants both empirically as well as theoretically. These investigations indicate that the sequential bootstrap is likely to perform better than the simple bootstrap in small to moderate samples. We present an expository account of these findings and discuss their potential applications.

Keywords: bootstrap, sequential resampling, poisson sampling, information content, sampling viewpoint, asymptotic correctness, empirical measures, empirical processes, weak convergence, quantiles

1. Introduction

The bootstrap is a ubiquitous technique to estimate performance characteristics of complex statistical methods which often do not admit simple closed form solutions. Its wide-spread use in statistics was initiated by Bradley Efron (1979), its rigorous mathematical foundations were established by Kesar Singh (1981) and its underlying principles have their origins in randomization, sample surveys, and Jackknife. In sample surveys terminology, its paradigm is akin to Mahalanobis' interpenetrating subsampling, as well as two-phase sampling (Hall, 2003). As an illustration, a two-phase sampling version of Efron's simple bootstrap can be paraphrased as follows. Consider a target population of finitely many units and for simplicity assume that its units take values in the real line. Let F , μ , and σ^2 respectively denote this population's distribution function, mean, and variance. Consider the problem of estimating these parameters. Let $\mathcal{S} = (X_1, X_2, \dots, X_n)$ be a random sample from this population. It is easily seen that δ_{X_1} is an unbiased estimator of F with δ_x being the unit probability function at x . Similarly X_1 is an unbiased estimator of μ , and $(X_1 - X_2)^2/2$ is an unbiased estimator of σ^2 . These estimators are evidently poor estimators. Given the observed sample \mathcal{S} , the Rao–Blackwellization of these estimators yields better estimators. These are:

$$F_n := E(\delta_{X_1} | \mathcal{S}) = n^{-1} \sum_{1 \leq i \leq n} \delta_{X_i}, \quad (1)$$

$$\bar{X}_n := E(X_1 | \mathcal{S}) = n^{-1} \sum_{1 \leq i \leq n} X_i, \quad (2)$$

$$s_n^2 := E((X_1 - X_2)^2/2 | \mathcal{S}) = (n-1)^{-1} \sum_{1 \leq i \leq n} (X_i - \bar{X}_n)^2. \quad (3)$$

The respective sampling errors of these estimators are:

$$(F_n - F), (\bar{X}_n - \mu), (s_n^2 - \sigma^2). \quad (4)$$

The simple bootstrap is a resampling technique for estimating such sampling errors and their distributions purely from the observed sample \mathcal{S} . In survey sampling

terminology, given \mathcal{S} , this entails a second phase of sampling in which one selects a with replacement simple random sample of size n from the sample \mathcal{S} from the first phase, resulting in the second phase sample:

$$\widehat{\mathcal{S}} := (\widehat{X}_1, \widehat{X}_2, \dots, \widehat{X}_n). \quad (5)$$

The survey sampling paradigm for two-phase sampling is to consider \mathcal{S} as a surrogate target population and the second-phase sample $\widehat{\mathcal{S}}$ as the surrogate sample \mathcal{S} . In terms of the surrogate target population \mathcal{S} and the surrogate sample $\widehat{\mathcal{S}}$, the estimators for the sampling errors in the preceding equations are given by

$$(\widehat{F}_n - F_n), (\widehat{\bar{X}}_n - \bar{X}_n), \left(\widehat{s}_n^2 - s_n^2 \right) \quad (6)$$

in which $\widehat{F}_n, \widehat{\bar{X}}_n, \widehat{s}_n^2$ are respectively the empirical distribution, mean, and variance based on the surrogate sample $\widehat{\mathcal{S}}$.

The conditional distributions of these statistics given \mathcal{S} furnish estimates of the sampling distribution of the corresponding statistics from phase one sampling.

Efron used repeated resamplings a large number of times (B) at the second phase to empirically estimate these distributions and referred to his method as “bootstrap resampling.” The main purpose of our illustration is to point out that Efron’s method has a great potential for extending his paradigm to more complex non-IID scenarios under the two-phase sampling framework with Rao–Blackwellization. For example, if the first-phase sampling is simple random sampling (without replacement) of size n from a population of size N then for the bootstrap sampling to go through, the second phase sampling needs to ensure that the surrogate image \widehat{F} of the underlying population F is such that for each unit in \widehat{F} , the probability of its inclusion in the second phase is $\approx (n/N)$ and for each pair of units in \widehat{F} , its probability of inclusion is $\approx n(n-1)/N(N-1)$.

What bootstrap is: There are numerous versions of the bootstrap. It has been studied for virtually all forms of statistical inference, including point and interval estimations, tests of hypothesis, classification models, and cross-validation (Chernick, 2008). For readers unfamiliar with the bootstrap, a brief outline of Efron’s simple bootstrap in the context of variance estimation is as follows:

1. Let F be a given population (distribution function) and let $\mu(F)$ be a given parameter to be estimated on the basis of an observed sample $\mathcal{S} := (X_1, X_2, \dots, X_n)$ drawn from the population F . Suppose that the plug-in estimator $\mu(F_n) = T(X_1, X_2, \dots, X_n)$, say, is used to estimate $\mu(F)$, in which F_n is the empirical distribution based on \mathcal{S} . Now to estimate the variance of $\mu(F_n)$, do.
 2. Draw observations $\widehat{X}_1, \widehat{X}_2, \dots, \widehat{X}_n$ from \mathcal{S} by simple random sampling with replacement (SRSWR), i.e., $(\widehat{X}_1, \widehat{X}_2, \dots, \widehat{X}_n) \sim F_n$.
 3. Compute $\widehat{T}_n = T(\widehat{X}_1, \widehat{X}_2, \dots, \widehat{X}_n)$.
 4. Repeat Steps 2 and 3, B times, say 1000, and let $(\widehat{T}_{n,1}, \widehat{T}_{n,2}, \dots, \widehat{T}_{n,B})$ denote the respective estimates so computed.
 5. The bootstrap estimate of the variance of $\mu(F_n)$ is

$$v_{\text{boot}} = \frac{1}{B} \sum_{j=1}^B \left(\widehat{T}_{n,j} - \frac{1}{B} \sum_{k=1}^B \widehat{T}_{n,k} \right)^2. \quad (7)$$

We turn now to examine Efron's simple bootstrap in some detail from the viewpoint of information content. We will continue to use the notations introduced in preceding discussions and suppose that the plug-in estimator $\mu(F_n)$ is to be used as an estimator of $\mu(F)$. Then, for a large class of functions μ , Efron's bootstrap resampling method (1979) provides a robust method of evaluating the performance characteristics of $\mu(F_n)$, solely on the basis of information derived (through randomization) from the observed sample \mathcal{S} . Consider the simple case when $\mu(F)$ is the population mean, $\mu(F_n)$ is the corresponding sample mean, and consider the pivot

$$\Pi_n = \sqrt{n}(\mu(F_n) - \mu(F))/\sqrt{\sigma^2(F_n)} \quad (8)$$

in which $\sigma^2(F_n)$ is the sample variance based on F_n .

The central limit theorem entails that the sampling distribution of Π_n can be approximated by the standard normal distribution. On the other hand, the bootstrap furnishes an alternative approach based on resampling to estimating the sampling distribution of Π_n from \mathcal{S} more precisely. Generally speaking, the central limit approximation is accurate to $o(1)$, while resampling approximation is accurate to $o(n^{-1/2})$. For example, let G_n denote the distribution function of Π_n . Then the central limit theorem yields that

$$\|G_n - \Phi\| := \sup_x |G_n(x) - \Phi(x)| = o(1), \quad (9)$$

where $\Phi(x)$ denotes the standard normal distribution function. The bootstrap approximation captures the skewness of the distribution G_n in the following sense:

$$\sqrt{n}\|G_n - H_n\| = o(1) \quad (10)$$

in which H_n represents a one-term Edgeworth expansion for G_n .

The simple bootstrap resampling scheme to approximating the distribution G_n is based on resampling by simple random sampling with replacement (SRSWR). Given the observed sample \mathcal{S} , let $\widehat{\mathcal{S}}_n = (\widehat{X}_1, \dots, \widehat{X}_n)$ be an SRSWR sample drawn from \mathcal{S} in this manner. Let \widehat{F}_n be the empirical distribution based on $\widehat{\mathcal{S}}_n$. Let

$$\widehat{\Pi}_n = \sqrt{n}(\mu(\widehat{F}_n) - \mu(F_n))/\sqrt{\sigma^2(\widehat{F}_n)} = \sqrt{n}(\widehat{\mu}_n - \mu_n)/\widehat{\sigma}_n, \quad \text{say}, \quad (11)$$

denote the pivot based on $\widehat{\mathcal{S}}_n$. Then for large n , the conditional distribution of $\widehat{\Pi}_n$ given \mathcal{S} is close to that of Π_n . In practice, this conditional distribution of $\widehat{\Pi}_n$ is simulated by repeated resampling from \mathcal{S} by SRSWR of size n , a large number of times usually of order $O(1000)$. This observed frequency distribution (ensemble) of $\widehat{\Pi}_n$ is referred to as the bootstrap distribution of the pivot Π_n . Thus for example, for large n , the quantiles from the bootstrap distribution of Π_n can be used to set up a confidence interval for μ based on the pivot Π_n .

Owing to the with replacement nature of SRSWR, not all of the observations in bootstrap samples $\widehat{\mathcal{S}}_n$, say, are based on distinct units from \mathcal{S} . In fact, the information content of $\widehat{\mathcal{S}}_n$, the set of observations from distinct units in $\widehat{\mathcal{S}}_n$, is a random variable. Let v_n denote the number of observations in $\widehat{\mathcal{S}}_n$ that are based on distinct units from \mathcal{S} . Then

$$E(v_n) = n \left[1 - \left(1 - \frac{1}{n} \right)^n \right] \simeq n(1 - e^{-1}) \simeq n(0.632), \quad (12)$$

$$\text{Var}(v_n) = \text{Var}(n - v_n) = n \left(1 - \frac{1}{n} \right)^n + n(n-1) \left(1 - \frac{2}{n} \right)^n - n^2 \left(1 - \frac{1}{n} \right)^{2n} \quad (13)$$

$$\simeq ne^{-1}(1 - e^{-1}). \quad (14)$$

In fact, the distribution of v_n approaches a binomial distribution $b(n,p)$ with $p \simeq 0.632$, showing that the information content of a bootstrap sample, is approximately sixty-three percent of the information content of the observed sample \mathcal{S} . In what follows, we describe alternatives to the simple bootstrap that keeps the information content of the bootstrap samples more stable.

2. A sequential bootstrap resampling scheme

Under this resampling scheme, we keep the information content of each bootstrap sample constant by requiring that the number of distinct observations in each sample be kept constant, equal to a preassigned value $\approx (0.632)n$ as follows:

To select a bootstrap sample, draw observations from \mathcal{S} sequentially by SRSWR until there are $(m+1) \approx n(1 - e^{-1}) + 1$ distinct observations in the observed bootstrap sample, the last observation in the bootstrap sample is discarded to ensure simplicity in technical details.

Thus an observed bootstrap sample has the form:

$$\widehat{\mathcal{S}}_N = (\widehat{X}_1, \widehat{X}_2, \dots, \widehat{X}_N) \quad (15)$$

in which $\widehat{X}_1, \dots, \widehat{X}_N$ have $m \approx n(1 - e^{-1})$ distinct observations from \mathcal{S} . The number of distinct observations in $\widehat{\mathcal{S}}_N$ is precisely $[n(1 - e^{-1})]$; it is no longer a random variable. The sample size N is a random variable with $E(N) \approx n$. The pivot based on $\widehat{\mathcal{S}}_N$ is

$$\widehat{\Pi}_N = \sqrt{N}(\mu(\widehat{F}_N) - \mu(F_n)) \quad (16)$$

in which \widehat{F}_N denotes the empirical distribution based on the bootstrap sample $\widehat{\mathcal{S}}_N$. For simplicity in exposition, we assume $\sigma^2(F) = 1$. Now for a comparative study of the pivot $\widehat{\Pi}_N$ based on the sequential resampling approach, versus $\widehat{\Pi}_n$ based on a bootstrap sample of fixed size n , it is necessary to estimate the order of magnitude of the random variable N . It is easily seen that N admits the following decomposition in terms of independent (geometric) random variables:

$$N = I_1 + I_2 + \dots + I_m \quad (17)$$

in which $m = [n(1 - e^{-1})]$, $I_1 = 1$ and for each k , $2 \leq k \leq m$, and for $j \geq 1$:

$$P(I_k = j) = \left(1 - \frac{k-1}{n}\right) \left(\frac{k-1}{n}\right)^{j-1}. \quad (18)$$

Therefore

$$E(N) = n \left[\frac{1}{n} + \frac{1}{(n-1)} + \cdots + \frac{1}{(n-m+1)} \right] = n + O(1) \quad (19)$$

since $m = [n(1 - e^{-1})]$.

Similarly, it is easily seen that

$$\text{Var}(N) = \sum_{k=1}^m \frac{n(k-1)}{(n-k+1)^2} = n(e-1) + O(1). \quad (20)$$

Thus

$$\frac{E(N-n)^2}{n^2} = \frac{(e-1)}{n} + O\left(\frac{1}{n^2}\right) \quad (21)$$

showing that $(N/n) \rightarrow 1$ in probability.

Further analysis similar to that of [Mitra and Pathak \(1984\)](#) can be used to show that

$$\frac{E(\widehat{\Pi}_N - \widehat{\Pi}_n)^2}{\text{Var}(\widehat{\Pi}_n)} \leq K \sqrt{\frac{\text{Var}(N)}{n^2}} = O\left(\frac{1}{\sqrt{n}}\right). \quad (22)$$

This implies that $\widehat{\Pi}_n$ and $\widehat{\Pi}_N$ are asymptotically equivalent. The preceding computations show that this sequential resampling plan, in addition to keeping the information content of bootstrap samples constant, also preserves its asymptotic correctness (consistency). In fact, our investigations show that under sequential forms of resampling, asymptotic correctness of the bootstrap procedure goes through only if the number m of distinct units in bootstrap samples satisfies: $m = n(1 - e^{-1}) + o(n)$. Consequently any other sequential resampling procedure for which first-order asymptotics goes through cannot be “significantly” different from ours in terms of its information content, e.g., consistency of variants of our sequential bootstrap by [Jiménez-Gamero et al. \(2004,2006\)](#), [Pino-Mejías et al. \(2010\)](#), and [Pauly \(2011\)](#) is a consequence of this regularity condition. Incidentally, the measure of disparity in two estimators as given by (22) is a simple tool frequently used in sampling theory to establish asymptotic equivalence of two competing estimators.

A second approach for consistency of the sequential bootstrap can be based on the so-called Mallows metric for weak convergence (cf [Bickel and Freedman, 1981](#)): In a given class of distributions with a finite second moment, define a given sequence of distributions $\{F_n\}$ to converge in M -sense to F if and only if (a) F_n converges weakly to F and (b) $\int x^2 dF_n$ converges to $\int x^2 dF$. It is easily seen that this notion of convergence is induced by the following M -metric:

$$d^2(F, G) = \inf_{X \sim F, Y \sim G} E(X - Y)^2 \quad (23)$$

in which the infimum is taken over all pairs of random variables (X, Y) with given marginals F and G respectively (Mallows, 1972).

Under this metric, (23), it is easily shown that

$$d(\widehat{\Pi}_n, \Pi_n) \leq d(F_n, F) \quad (24)$$

so that

$$d(\widehat{\Pi}_n, \Phi) \leq d(\widehat{\Pi}_n, \Pi_n) + d(\Pi_n, \Phi) \leq d(F_n, F) + d(\Pi_n, \Phi). \quad (25)$$

The first term on the right converges to zero by the law of large numbers (the Glivenko–Cantelli lemma), while the second term does so by the central limit theorem. This establishes the consistency of the simple bootstrap. An added complication that arises in sequential resampling scheme is that the bootstrap sample size N is now a random variable. Thus for the consistency of the sequential bootstrap to go through, one needs to show that $d(\widehat{\Pi}_N, \Phi)$ can be made arbitrarily small; (24) is no longer directly applicable. Nevertheless, based on techniques similar to those of Pathak (1964) and Mitra and Pathak (1984) (Lemma 3.1 in Mitra and Pathak), it can be shown that:

$$d(\widehat{\Pi}_N, \widehat{\Pi}_n) = O(n^{-1/4}). \quad (26)$$

Consistency of the pivot $\widehat{\Pi}_N$ follows from (24) and (26) and the triangle inequality. A limitation of the preceding two approaches is that they apply only to linear statistics and cannot be easily extended to more general statistics. A third and a more general approach is to treat pivots like $\widehat{\Pi}_n$ and $\widehat{\Pi}_N$ as random signed measures and study their convergence in the functional sense. In this functional setting, we now describe the key result which furnishes a rigorous justification of the sequential bootstrap for a large class of functionals.

3. Bootstrapping empirical measures with a random sample size

For simplicity in exposition, we tacitly assume at the outset the existence of a suitable probability space where the random variables, functions, etc., under study are well-defined (see Rao et al., 1997). Now consider $\{X_n : n \geq 1\}$ a sequence of independent random elements with a common distribution P in a certain given space χ . Let P_n denote the empirical measure based on the sample (X_1, \dots, X_n) from P , i.e.,

$$P_n := n^{-1} \sum_{1 \leq i \leq n} \delta_{X_i} \quad (27)$$

with δ_x being the unit point mass at $x \in X$.

Let $\{\widehat{X}_{n,j} : j \geq 1\}$ be a sequence of independent random elements with common distribution P_n . We refer to this sequence as a sequence of bootstrapped observations, or just a bootstrapped sequence. Given a number $N \geq 1$, let $\widehat{P}_{n,N}$ denote the empirical measure based on the bootstrap sample $(\widehat{X}_{n,1}, \dots, \widehat{X}_{n,N})$ of size N ; i.e.,

$$\widehat{P}_{n,N} := N^{-1} \sum_{1 \leq i \leq N} \delta_{\widehat{X}_{n,i}}. \quad (28)$$

We refer to it as a bootstrapped empirical measure of size N . The main object of this section is to show that if (N/n) converges to one in probability, then the bootstrapped empirical measure $\widehat{P}_{n,N}$ is at a distance of $o(n^{-1/2})$ from the bootstrapped empirical measure $\widehat{P}_n := \widehat{P}_{n,n}$, so that all of the \sqrt{n} -asymptotic results for the classical bootstrap carry over to the sequential bootstrap with a random sample size.

To do so, define the random measure: $Z_n := \sqrt{n}(P_n - P), n \geq 1$, and for any class \mathcal{F} of measurable functions $f : \chi \mapsto R^1$, let G_P be a mean zero Gaussian process indexed by \mathcal{F} , with covariance $E[f(X)g(X)] - Ef(X)Eg(X)$ for all $f, g \in \mathcal{F}$. Both Z_n and G_P can be viewed as being indexed by \mathcal{F} . We say that \mathcal{F} is P-Donsker if Z_n converges in the Hoffman-Jørgensen weak sense to G_P in the space $\ell^\infty(\mathcal{F})$ of all uniformly bounded functions on the class \mathcal{F} . In this case we say that $\mathcal{F} \in CLT(P)$. The following result is well known (Giné and Zinn, 1986). If $\mathcal{F} \in CLT(P)$, then

$$E^* \left\| \sum_{1 \leq i \leq n} (\delta_{X_i} - P) \right\|_{\mathcal{F}} = O(n^{1/2}) \quad (29)$$

in which E^* is the so-called outer expectation.

The investigation of the classical bootstrap for the general empirical measures was initiated by Gaensler (1987). A remarkable theorem due to Giné and Zinn (1990) is the equivalence of the following two conditions:

- (a) $\mathcal{F} \in CLT(P)$.
- (b) There exists a Gaussian process G_P defined on a certain probability space such that

$$\sqrt{n}(\widehat{P}_n - P_n) \rightarrow G_P \quad \text{weakly in } \ell^\infty(\mathcal{F}) \quad (30)$$

establishing the equivalence of the consistency of the simple bootstrap for the general empirical measures over a class \mathcal{F} and the corresponding central limit theorem: $\mathcal{F} \in CLT(P)$. Praestgaard and Wellner (1993) studied more general versions of the bootstrap, including the one in which the size of bootstrap sample $m \neq n$.

We adopt the following terminology in the sequel. Given a sequence $\{\eta_n : n \geq 1\}$ of random variables and a sequence $\{a_n : n \geq 1\}$ of positive real numbers, we write

$$\eta_n = o_p(a_n) \quad (31)$$

iff, for each $\varepsilon > 0$,

$$\limsup_{n \rightarrow \infty} E^*(I(|\eta_n| \geq \varepsilon a_n)) = 0. \quad (32)$$

We write

$$\eta_n = O_p(a_n) \quad (33)$$

iff

$$\lim_{c \rightarrow \infty} \limsup_{n \rightarrow \infty} E^*(I(|\eta_n| \geq ca_n)) = 0. \quad (34)$$

It is easy to check that $\eta_n = O_p(a_n)$ if and only if $\eta_n = o_p(b_n)$, for any sequence b_n of positive numbers such that $a_n = o(b_n)$. An immediate consequence of Fubini's theorem is that $\eta_n = o_p(a_n)$ iff, for each $\varepsilon > 0$,

$$\mathbf{I}(|\eta_n| \geq \varepsilon a_n) = o_p(1), \quad (35)$$

i.e., the random element on the left-hand side of (35) converges to zero in probability.

We now turn to the key results of this section that are instrumental in establishing consistency of various forms of sequential bootstrap resampling schemes in the literature.

Theorem 1 *Let $\{a_n\}$ be a sequence of positive real numbers such that $a_n = O(n)$. Suppose that $\mathcal{F} \in CLT(P)$ and let*

$$|N_n - n| = o_p(a_n). \quad (36)$$

Then

$$|\widehat{P}_{n,N_n} - \widehat{P}_n|_{\mathcal{F}} = o_p\left(\frac{a_n^{1/2}}{n}\right). \quad (37)$$

The proof of this theorem follows mainly from the Giné–Zinn theorem (1990), (30) and suitable triangle inequalities (cf Rao et al. (1997) for details).

Corollary 1 *Let $\{a_n\}$ be a sequence of positive numbers such that $a_n = o(n)$. Suppose that $\mathcal{F} \in CLT(P)$ and $|N_n - n| = O_p(a_n)$. Then*

$$|\widehat{P}_{n,N_n} - \widehat{P}_n|_{\mathcal{F}} = O_p\left(\frac{a_n^{1/2}}{n}\right). \quad (38)$$

For a proof, apply Theorem 1 to any sequence $\{b_n\}$ such that $a_n = o(b_n)$.

Corollary 2 *Suppose that $\mathcal{F} \in CLT(P)$ and $|N_n - n| = o_p(n)$. Then*

$$|\widehat{P}_{n,N_n} - \widehat{P}_n|_{\mathcal{F}} = o_p(n^{-1/2}). \quad (39)$$

Corollary 3 *Under the conditions of Corollary 2,*

$$\left| N_n^{1/2} (\widehat{P}_{n,N_n} - P_n) - n^{1/2} (\widehat{P}_n - P_n) \right|_{\mathcal{F}} = o_p(1). \quad (40)$$

Moreover, under the conditions of Theorem 1,

$$\left| N_n^{1/2} (\widehat{P}_{n,N_n} - P_n) - n^{1/2} (\widehat{P}_n - P_n) \right|_{\mathcal{F}} = o_p\left(\sqrt{\frac{a_n}{n}}\right) \quad (41)$$

and under the conditions of Corollary 2

$$\left\| N_n^{1/2}(\widehat{P}_{n,N_n} - P_n) - n^{1/2}(\widehat{P}_n - P_n) \right\|_{\mathcal{F}} = O_p \left(\sqrt{\frac{a_n}{n}} \right). \quad (42)$$

Indeed,

$$\left\| N_n^{1/2}(\widehat{P}_{n,N_n} - P_n) - n^{1/2}(\widehat{P}_n - P_n) \right\|_{\mathcal{F}}, \quad (43)$$

$$\leq |N_n^{1/2} - n^{1/2}| (\|\widehat{P}_{n,N_n} - P_n\|_{\mathcal{F}} + \|\widehat{P}_n - P_n\|_{\mathcal{F}}) \\ + n^{1/2} \|\widehat{P}_{n,N_n} - \widehat{P}_n\|_{\mathcal{F}}. \quad (44)$$

Under the conditions of Corollary 2,

$$\left| N_n^{1/2} - n^{1/2} \right| = n^{1/2} \left| \left(\frac{N_n}{n} \right)^{1/2} - 1 \right| = o_p(n^{1/2}). \quad (45)$$

Under the conditions of Theorem 1 we have

$$\left| N_n^{1/2} - n^{1/2} \right| = o_p(a_n n^{-1}) = o_p\left(\sqrt{\frac{a_n}{n}}\right). \quad (46)$$

The Giné–Zinn Theorem implies that $\{n^{1/2} \|\widehat{P}_n - P_n\|_{\mathcal{F}}\}$ is stochastically bounded, and the result follows from Corollary 2 and Theorem 1.

4. Convergence rates for the sequential bootstrap

In what follows we apply the results of Section 3 to the empirical measures based on the sequential bootstrap. We start with the following theorem summarizing the properties of the empirical measures in this case.

Theorem 2 Suppose that $\mathcal{F} \in CLT(P)$ and let \widehat{P}_{n,N_n} be the empirical measure based on a sequential bootstrap sample. Then

$$|\widehat{P}_{n,N_n} - \widehat{P}_n|_{\mathcal{F}} = O_p(n^{-3/4}) \quad (47)$$

and

$$\left| N_n^{1/2}(\widehat{P}_{N_n,n} - P_n) - n^{1/2}(\widehat{P}_n - P_n) \right|_{\mathcal{F}} = O_p(n^{-1/4}). \quad (48)$$

Indeed, in this case we have, by (21), $|N_n - n| = O_p(n^{1/2})$ and Corollaries 1 and 3 imply the result. In fact, a more careful analysis, e.g., by invoking the Mills ratio, allows one to replace O -terms by o -terms in these equations. It is worth noting that Eq. (48) here implies the asymptotic equivalence of the sequential bootstrap and the simple bootstrap.

Illustrative Example: Consider the case $X = R^1$ and let $\mathcal{F} := \{I_{(-\infty,t]} : t \in R^1\}$ so that general empirical measures considered earlier turn out to be the classical empirical distribution functions. Denote

$$F(t) := P((-\infty, t]) = \int_{R^1} I_{(-\infty, t]} dP, \quad (49)$$

$$F_n(t) := P_n((-\infty, t]) = \int_{R^1} I_{(-\infty, t]} dP_n, \quad (50)$$

and

$$\widehat{F}_{n,N}(t) := \widehat{P}_{n,N}((-\infty, t]) = \int_{R^1} I_{(-\infty, t]} d\widehat{P}_{n,N}. \quad (51)$$

We also use the abbreviation $\widehat{F}_n := \widehat{F}_{n,n}$ and $|\cdot|_\infty$ denotes the sup-norm.

Since, by the Kolmogorov-Donsker theorem, $\mathcal{F} \in CLT(P)$ for all Borel probability measures P on R^1 (where $\mathcal{F} = \{I_{(-\infty, t]} : t \in R^1\}$), we have the following result.

Theorem 3 *For any distribution function F on R^1 ,*

$$|\widehat{F}_{n,N_n} - \widehat{F}_n|_\infty = O_p(n^{-3/4}) \quad (52)$$

and

$$\left| N_n^{1/2}(\widehat{F}_{n,N_n} - F_n) - n^{1/2}(\widehat{F}_n - F_n) \right|_\infty = O_p(n^{-1/4}). \quad (53)$$

In case F is the uniform distribution on $[0,1]$, we get as a trivial corollary that the sequence of stochastic processes

$$\left\{ N_n^{1/2}(\widehat{F}_{n,N_n}(t) - F_n(t)) : t \in [0,1] \right\}_{n \geq 1} \quad (54)$$

converges weakly (in the space $\ell^\infty([0,1])$ or $D[0,1]$) to the standard Brownian bridge process

$$B(t) := w(t) - tw(1), \quad t \in [0,1], \quad (55)$$

w being the standard Wiener process. More generally, if F is continuous, then the limit is the process $(B \circ F)(t) = B(F(t))$, $t \in R^1$. These facts easily imply justification of the sequential bootstrap for a variety of statistics θ_n , which can be represented as $\theta_n = T(F_n)$ with a compactly (Hadamard) differentiable functional T . More precisely, let T be a functional (or, more generally, an operator with values in a linear normed space) $G \mapsto T(G)$, defined on a set \mathcal{G} of distribution functions G . Then T is supposed to be compactly differentiable at F tangentially to the space of all uniformly bounded and uniformly continuous functions (this is differentiability in a certain uniform sense and is much like uniform continuity on compact sets (cf Gill, 1989)). For such statistics, we have

$$T(\widehat{F}_{n,N_n}) - T(\widehat{F}_n) = o_p(n^{-1/2}), \quad (56)$$

proving the first-order asymptotic correctness of the sequential resampling approach. These observations can be applied, for instance, to the operator $G \mapsto G^{-1}$, defined by

$$G^{-1}(t) := \inf\{x \in R^1 : G(x) \geq t\} \quad (57)$$

and taking a distribution function to its quantile function (see, e.g., *Fernholz (1983)* or *Gill (1989)* for compact differentiability of such operators). Specifically, if F is continuously differentiable at a point $x \in R^1$ with $F'(x) > 0$, then

$$\left| \widehat{F}_{n,N_n}^{-1}(t) - \widehat{F}_n^{-1}(t) \right| = o_p(n^{-1/2}), \quad (58)$$

where $t = F(x)$. In fact, we have obtained bounds for quantiles that are sharper than (58) (see Theorem 4.4 in *Rao et al., 1997*). The different approaches that we have described so far show that the distance between the sequential and the usual bootstrap is at most of the order of $O(n^{-\frac{1}{4}})$. Although this entails consistency of the sequential bootstrap, it does not guarantee its second-order correctness. We turn now to two approaches to establish second-order correctness of the sequential bootstrap.

5. Second-order correctness of the sequential bootstrap

The proof of the second-order correctness of the sequential bootstrap requires the Edgeworth expansion for dependent random variables. Along the lines of the **Hall-Mammen work (1994)**, we first outline an approach based on cumulants. This approach assumes that a formal Edgeworth expansion is valid for pivot under the sequential bootstrap.

Let N_i denote the number of times the i th observation x_i from the original sample appears in the sequential bootstrap sample, $1 \leq i \leq n$. Then

$$N = N_1 + N_2 + \cdots + N_n \quad (59)$$

in which N_1, N_2, \dots are exchangeable random variables.

The probability distribution of N is given by

$$P(N = k) = \binom{n-1}{m} \Delta^m \left(\frac{x}{n}\right)^k \quad (60)$$

for $k \geq m$, and in which Δ is the difference operator with unit increment. The moment generating function of N is given by

$$M(t) = E(e^{tN}) = \binom{n-1}{m} \Delta^m \frac{n}{(n - xe^t)}. \quad (61)$$

The second-order correctness of the sequential bootstrap for linear statistics such as the sample sum is closely related to the behavior of the moments of the random variables $\{N_i : 1 \leq i \leq n\}$. Among other things, the asymptotic distribution of each N_i is Poisson with mean 1. In fact, it can be shown that

$$E(N_1 - 1)^{k_1} \cdots (N_i - 1)^{k_i} = \prod_{j=1}^i (e^\Delta - 1 - \Delta)(x - 1)^{k_j} + O\left(\frac{1}{n}\right). \quad (62)$$

It follows from (62) that to order $O(n^{-1})$, the random variables $\{N_i : 1 \leq i \leq n\}$ are asymptotically independent. This implies that the Hall-Mammen-type (1994) conditions for the second-order correctness of the sequential bootstrap hold. This approach is based on the tacit assumption that formal Edgeworth-type expansions go through for the sequential bootstrap. A rigorous justification of such an approach is unavailable in the literature at the present time. Another approach which bypasses this difficulty altogether entails a slight modification of the sequential bootstrap. It is based on the observation that each N_i in Eq. (59) is approximately a Poisson variate subject to the constraint:

$$I(N_1 > 0) + I(N_2 > 0) + \cdots + I(N_n > 0) = m \approx n(1 - e^{-1}) \quad (63)$$

i.e., there are exactly m non-zero N_i s, $1 \leq i \leq n$. This observation enables us to modify the sequential bootstrap so that existing techniques on the Edgeworth expansion, such as those of Babu and Bai (1996), Bai and Rao (1991, 1992), Babu and Singh (1989), and others, can be employed. We refer to this modified version as the Poisson bootstrap.

The Poisson Bootstrap: The original sample (x_1, \dots, x_n) is assumed to be from \mathbb{R}^k for greater flexibility. Let $\alpha_1, \dots, \alpha_n$ denote n independent observations from $P(1)$, the Poisson distribution with unit mean. If there are exactly $m = [n(1 - e^{-1})]$ non-zero values among $\alpha_1, \dots, \alpha_n$, take

$$\widehat{S}_N = \{(x_1, \alpha_1), \dots, (x_n, \alpha_n)\} \quad (64)$$

otherwise reject the α s and repeat the procedure. This is the conceptual definition. The sample size N of the Poisson bootstrap admits the representation:

$$N = \alpha_1 + \alpha_2 + \cdots + \alpha_n \quad (65)$$

in which $\alpha_1, \dots, \alpha_n$ are IID Poisson variates with mean = 1 and with the added restriction that exactly m of the n α s are non-zero, i.e., $I(\alpha_1 > 0) + \cdots + I(\alpha_n > 0) = m$.

A simple way to implement the Poisson bootstrap in practice is to first draw a simple random sample without replacement (SRSWOR) of size m from the set of unit-indices $\{1, 2, \dots, n\}$, say (i_1, i_2, \dots, i_m) . Then assign respectively to these $\alpha_{i_1}, \alpha_{i_2}, \dots, \alpha_{i_m}$ values independently drawn from the truncated Poisson distribution with $\lambda = 1$ and left-truncated at $x = 0$ (R-syntax: qpois (runif (m, dpois (0,1), 1), 1)) and set $\alpha = 0$ for the remaining α s.

It can be shown that the moment generating function $M_N(t)$ of $N = \alpha_1 + \alpha_2 + \cdots + \alpha_n$ is (Theorem 2.1 in Babu et al. (1999)):

$$M_N(t) = \left[\frac{(e^{(e^t-1)} - e^{-1})}{(1 - e^{-1})} \right]^m \quad (66)$$

so that the distribution of N can be viewed as that of m IID random variables with a common moment generating function:

$$m(t) = \frac{(e^{(e^t-1)} - e^{-1})}{(1 - e^{-1})}. \quad (67)$$

It is clear that $m(t)$ is the moment generating function of the Poisson distribution with location parameter $\lambda = 1$ and truncated at $x = 0$.

This modification of the sequential bootstrap enables us to develop a rigorous proof of the second-order correctness in the sequential case. Now let (X_1, \dots, X_n) be IID random variables with mean μ and variance σ^2 . We assume that X_1 is strongly non-lattice, i.e., it satisfies Cramér's condition:

$$\limsup_{|t| \rightarrow \infty} |E(\exp(itX_1))| < 1. \quad (68)$$

Let $\{Y_j : j \geq 1\}$ be a sequence of IID Poisson random variables with mean 1. We now state three main results, furnishing a rigorous justification for the second-order correctness of the sequential bootstrap. These results follow from conditional Edgeworth expansions for weighted means of multivariate random vectors (cf Babu et al., 1999).

Theorem 4 Suppose that $E\|X_1\|^5 < \infty$ and that the characteristic function of X_1 satisfies Cramér's condition (68). If $m - n(1 - e^{-1})$ is bounded, then

$$\begin{aligned} & P\left(\frac{1}{\sqrt{N}} \sum_{i=1}^n (X_i - \bar{X}) Y_i \leq xs_n \mid T_n = m; X_1, \dots, X_n\right) \\ & \quad - P\left(\frac{1}{\sqrt{n}} \sum_{i=1}^n (X_i - E(X_1)) \leq x\sigma\right) \\ & = O_p(n^{-1}) \end{aligned} \quad (69)$$

uniformly in x , given $T_n := \sum_{i=1}^n I(Y_i > 0) = m$.

Smooth Functional Model: An extension of Theorem 4 to the multivariate case goes through to statistics which can be expressed as smooth functions of multivariate means. Now let X_1, \dots, X_n be a sequence of IID random vectors with mean μ and dispersion matrix Σ . Let Σ_n denote the corresponding sample dispersion matrix. Then the following results hold.

Theorem 5 Suppose that X_1 is strongly non-lattice and $E\|X_1\|^4 < \infty$. Let H be a three-times continuously differentiable function in a neighborhood of μ . Let $l(y)$ denote the vector of first-order partial derivatives at y and suppose that $l(\mu) \neq 0$. If $m - n(1 - e^{-1})$ is bounded, then for almost all sample sequences $\{X_i : 1 \leq i \leq n\}$, we have

$$\begin{aligned} & \sqrt{n} \left| P\left(\frac{\sqrt{N}(H(N^{-1} \sum_{i=1}^n X_i Y_i) - H(\bar{X}_n))}{\sqrt{l'(\bar{X}_n) \Sigma_n l(\bar{X}_n)}} \right. \right. \\ & \quad \left. \left. \leq x \mid T_n = m; X_1, \dots, X_n \right) \right. \\ & \quad - P\left(\frac{\sqrt{n}(H(\bar{X}_n) - H(\mu))}{\sqrt{l'(\mu) \Sigma l(\mu)}} \leq x \right) \Big|_\infty = o(1) \end{aligned} \quad (70)$$

in which $|\cdot|_\infty$ denotes the sup-norm over x .

The following result is well suited for applications to studentized statistics.

Theorem 6 Let $\{X_i : 1 \leq i \leq n\}$ satisfy the conditions of Theorem 5. Suppose that the function H is three-times continuously differentiable in the neighborhood of the origin and $H(0) = 0$. If $m - n(1 - e^{-1})$ is bounded, then for almost all sample sequences $\{X_i : 1 \leq i \leq n\}$, we have

$$\begin{aligned} & \sqrt{n} \left| P \left(\frac{\sqrt{N}(H(N^{-1} \sum_{i=1}^n (X_i - \bar{X}) Y_i))}{\sqrt{l'(0) \Sigma l(0)}} \right. \right. \\ & \quad \left. \left. \leq z | T_n = m; X_1, \dots, X_n \right) \right. \\ & \quad \left. - P \left(\frac{\sqrt{n}(H(\bar{X}_n - \mu))}{\sqrt{l'(0) \Sigma l(0)}} \leq z \right) \right|_{\infty} = o(1) \end{aligned} \quad (71)$$

For example, an immediate consequence of Theorem 6 is the second-order correctness of the following sequential bootstrap pivot:

$$\hat{\pi}_N = \sqrt{N} \left(\sum_{i=1}^n (X_i - \bar{X}) Y_i \right) / s_n \quad (72)$$

given that $T_n := \sum_{i=1}^n I(Y_i > 0) = m$.

6. Concluding remarks

The sequential bootstrap discussed here has its origins in the simple bootstrap of Bradley Efron and his observation that the information content of the simple bootstrap is supported on approximately $0.632n$ of the original data. It is well known that owing to the with replacement nature of resampling in the simple bootstrap, the information content of its bootstrapped sample is a random variable. The sequential bootstrap keeps the information content of its bootstrapped samples at $0.632n$ -level of the original sample size. This constancy of information in the sequential bootstrap samples ensures homogeneity of information and reduces the likelihood of information-deficient samples. The sequential bootstrap provides better breakdown points. The empirical characteristics of the sequential bootstrap are within $O(n^{-3/4})$ of the simple bootstrap and its second-order correctness goes through under existing regularity conditions. It is perhaps worthwhile to conclude this article with a final remark that recent studies in the literature (Pino-Mejías et al., 2010; Pauly (2011), and others) indicate that the sequential bootstrap is likely to perform better than the simple bootstrap in small to moderate size samples under a variety of applications commonly encountered in practice.

Acknowledgments

The authors are grateful to Professors Babu, Koltchinskii, and Koul for their perusal and helpful remarks. This work was supported in part from NIH Grants RC065888, RC060667, and RC060042.

References

- Babu, G.J., Bai, Z.D., 1996. Mixture of global and local Edgeworth expansions and their applications. *J. Multivar. Anal.* 59, 282–307.
- Babu, G.J., Singh, K., 1989. On Edgeworth expansions in the mixture cases. *Ann. Stat.* 17, 443–447.
- Babu, G.J., Pathak, P.K., Rao, C.R., 1999. Second-order correctness of the Poisson bootstrap. *Ann. Stat.* 27, 1666–1683.
- Bai, Z.D., Rao, C.R., 1991. Edgeworth expansions of a function of sample means. *Ann. Stat.* 19, 1295–1315.
- Bai, Z.D., Rao, C.R., 1992. A note on the Edgeworth expansion for ratio of sample means. *Sankhyā A* 19, 309–322.
- Bickel, P.J., Freedman, D.A., 1981. Some asymptotic theory for the bootstrap. *Ann. Stat.* 9, 1196–1217.
- Chernick, M.R., 2008. *Bootstrap Methods: A Guide for Practitioners and Researchers*, second ed. Wiley, Hoboken, New Jersey.
- Efron, B., 1979. Bootstrap methods: another look at the jackknife. *Ann. Stat.* 7, 1–26.
- Efron, B., 1983. Estimating the error rate of a prediction rule: improvement on cross-validation. *J. Am. Stat. Assoc.* 78, 316–331.
- Fernholz, L.T., 1983. Von Mises Calculus for Statistical Functionals. Lecture Notes in Statistics, vol. 19. Springer-Verlag, New York.
- Gaenssler, P., 1987. Bootstrapping empirical measures indexed by Vapnik–Červonenkis classes of sets. In: Prohorov, Yu. V., Statulevicius, V.A., Sazonov, V., et al. (Eds.), *Probability Theory and Mathematical Statistics*, vol. 1. VNU Science Press, Utrecht, pp. 467–481.
- Gill, R., 1989. Non- and semi-parametric maximum likelihood estimators and the von Mises method (Part 1). *Scand. J. Stat.* 16, 97–128.
- Giné, E., Zinn, J., 1986. Lectures on the Central Limit Theorem for Empirical Processes. Lecture Notes in Mathematics, vol. 1221. Springer, pp. 50–113.
- Giné, E., Zinn, J., 1990. Bootstrapping general empirical measures. *Ann. Probab.* 18, 851–869.
- Hall, P., 2003. A short prehistory of the bootstrap. *Stat. Sci.* 18, 158–167.
- Hall, P., Mammen, E., 1994. On general resampling algorithms and their performance in distribution estimation. *Ann. Stat.* 24, 2011–2030.
- Jiménez-Gamero, M.D., Muñoz-García, J., Pino-Mejías, R., 2004. Reduced bootstrap for the median. *Stat. Sinica* 14, 1179–1198.
- Jiménez-Gamero, M.D., Muñoz-García, J., Cubiles-de-la-Vega, M.D., 2006. Consistency of the reduced bootstrap for sample means. *Stat. Probab. Lett.* 76, 689–697.
- Mallows, C.L., 1972. A note on asymptotic joint normality. *Ann. Math. Stat.* 43, 508–515.
- Mammen, E., 1992. *When Does Bootstrap Work?* Springer-Verlag, New York.
- Mitra, S.K., Pathak, P.K., 1984. The nature of simple random sampling. *Ann. Stat.* 12, 1536–1542.
- Pathak, P.K., 1964. Sufficiency in sampling theory. *Ann. Math. Stat.* 35, 795–808.
- Pathak, P.K., 1964. On inverse sampling with unequal probabilities. *Biometrika* 51, 185–193.
- Pauly, M., 2011. Consistency of the subsample bootstrap empirical process. *Stat. iFirst* 45, 1–6.
- Pino-Mejías, R., Jiménez-Gamero, M.D., González, A.E., 2010. A Monte Carlo comparison of three consistent bootstrap procedures. Personal Communication.
- Praestgaard, J., Wellner, J., 1993. Exchangeably weighted bootstraps of the general empirical process. *Ann. Probab.* 21, 2053–2086.
- Rao, C.R., Pathak, P.K., Koltchinskii, V.I., 1997. Bootstrap by sequential resampling. *J. Stat. Plan. Infer.* 64, 257–281.
- Shoemaker, O.J., Pathak, P.K., 2001. The sequential bootstrap: a comparison with regular bootstrap. *Commun. Stat. Theory Methods* 30, 1661–1674.
- Singh, K., 1981. On the asymptotic accuracy of Efron's bootstrap. *Ann. Stat.* 9, 1187–1195.

The Cross-Entropy Method for Estimation

Dirk P. Kroese¹, Reuven Y. Rubinstein², and Peter W. Glynn³

¹*School of Mathematics and Physics, The University of Queensland,
Brisbane 4072, Australia*

²*Faculty of Industrial Engineering and Management, Technion, Haifa, Israel*

³*Department of Management Science and Engineering, Stanford University,
CA, USA*

Abstract

This chapter describes how difficult statistical estimation problems can often be solved efficiently by means of the *cross-entropy* (CE) method. The CE method can be viewed as an adaptive importance sampling procedure that uses the cross-entropy or Kullback–Leibler divergence as a measure of closeness between two sampling distributions. The CE method is particularly useful for the estimation of rare-event probabilities. The method can also be used to solve a diverse range of optimization problems. The optimization setting is described in detail in the chapter entitled “The Cross-Entropy Method for Optimization.”

Keywords: cross-entropy, estimation, rare events, importance sampling, adaptive Monte Carlo, zero-variance distribution

1. Introduction

The CE method was introduced by Rubinstein (1999, 2001), extending earlier work on variance minimization (Rubinstein, 1997). Originally, the CE method was developed as a means of computing rare-event probabilities; that is, very small probabilities—say less than 10^{-4} . Naive Monte Carlo estimation of such a probability requires a large simulation effort, inversely proportional to the magnitude of the rare-event probability. The CE method is based on two ideas. The first idea is to estimate the probability of interest by gradually changing the sampling distribution, from the original to a distribution for which the rare event is much more likely to happen. To remove the estimation bias, importance sampling is used. The second idea is to use the CE distance to construct the sequence of sampling

distributions. This significantly simplifies the numerical computation at each step, and provides fast and efficient algorithms that are easy to implement by practitioners.

The CE method has been successfully applied to a diverse range of estimation and optimization problems. References on estimation include [Asmussen et al. \(2005\)](#), [de Boer \(2000\)](#), [de Boer et al. \(2004\)](#), [Chan and Kroese \(2010, 2012\)](#), [Chan et al. \(2011\)](#), [Hui et al. \(2005\)](#), [Homem-de-Mello \(2007\)](#), [Kroese and Hui \(2006\)](#), [Kroese and Rubinstein \(2004\)](#), [Rao \(2010\)](#), and [Ridder \(2005\)](#). Parallel implementations of the CE method are discussed in [Evans \(2009\)](#) and [Evans et al. \(2007\)](#), and other generalizations and advances are explored in [Taimre \(2009\)](#). For many more references on optimization we refer to the accompanying chapter in this handbook on CE optimization.

A tutorial on the CE method is given in [de Boer et al. \(2005\)](#). A comprehensive treatment can be found in [Rubinstein and Kroese \(2004\)](#); see also [Rubinstein and Kroese \(2007, Chapter 8\)](#) and [Kroese et al. \(2011, Chapter 13\)](#). The CE method homepage is www.cemethod.org.

2. Estimation setting

The general setting of the CE method concerns the estimation of an expectation ℓ of the form

$$\ell = \mathbb{E}_f[H(\mathbf{X})] = \int H(\mathbf{x})f(\mathbf{x})d\mathbf{x}, \quad (1)$$

where H is a real-valued function and f is the probability density function (pdf) of a random variable (possibly multi-dimensional) \mathbf{X} . For simplicity it is assumed that \mathbf{X} is a continuous random variable. For the discrete case, replace the integral in (1) with a sum. Of particular importance is the case where $H(\mathbf{x}) = \mathbf{I}_{\{S(\mathbf{x}) \geqslant \gamma\}}$, where S is another real-valued function (here and for the rest of the chapter \mathbf{I}_A denotes an indicator function of an event A). This special case is further discussed in Section 2.4.

The *crude Monte Carlo* (CMC) estimator of ℓ is

$$\hat{\ell}_{\text{CMC}} = \frac{1}{N} \sum_{k=1}^N H(\mathbf{X}_k),$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N$ are independent and identically distributed (iid) with density f . We write $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f$. It is not difficult to see (see, for example, [Rubinstein and Kroese \(2007, Page 27\)](#)) that the *relative error* of the CMC estimator is given by

$$\text{RE}_{\text{CMC}} \stackrel{\text{def}}{=} \frac{\sqrt{\text{Var}(\hat{\ell}_{\text{CMC}})}}{\ell} = \frac{1 - \ell}{N\ell} \approx \sqrt{\frac{1}{N\ell}}.$$

For example, to obtain a relative error of 1% for a probability of $\ell = 10^{-6}$, the sample size N needs to be 10^{10} .

The idea behind *importance sampling* is to sample the $\{\mathbf{X}_i\}$ from a different pdf g , under which the rare event may be more likely, while compensating for the bias

thus introduced. In particular, let g be a pdf for which $H(\mathbf{x})f(\mathbf{x}) \neq 0$ for every \mathbf{x} . The expectation ℓ can be written as

$$\ell = \int H(\mathbf{x}) \frac{f(\mathbf{x})}{g(\mathbf{x})} g(\mathbf{x}) d\mathbf{x} = \mathbb{E}_g \left[H(\mathbf{X}) \frac{f(\mathbf{X})}{g(\mathbf{X})} \right], \quad (2)$$

where the subscript g indicates that the expectation is taken with respect to g rather than f . Consequently, if $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} g$, then

$$\hat{\ell} = \frac{1}{N} \sum_{k=1}^N H(\mathbf{X}_k) \frac{f(\mathbf{X}_k)}{g(\mathbf{X}_k)} \quad (3)$$

is an unbiased estimator of ℓ . The quotient $W(\mathbf{X}) \stackrel{\text{def}}{=} f(\mathbf{X})/g(\mathbf{X})$ is called the *likelihood ratio* of \mathbf{X} . Estimating ℓ via such an *importance sampling* estimator may be beneficial, in terms of a smaller relative error for the same simulation effort, if the importance sampling pdf g is chosen appropriately. Note that by taking $f = g$ one obtains the CMC estimator.

2.1. Variance minimization

The optimal importance sampling pdf, that is, the pdf g^* for which the variance of $\hat{\ell}$ is minimal, is proportional to $|H|f$ (see, e.g., [Rubinstein and Kroese \(2007, Page 132\)](#)). Indeed, if H is strictly positive then under $g^* \propto Hf$ the importance sampling estimator has *zero variance*. However, g^* is in general difficult to evaluate; for example, when $H \geq 0$ the normalization constant of g^* is precisely ℓ . Often the “nominal” pdf f can be embedded in a parameterized class of densities $\{f(\cdot; \mathbf{v})\}$, where f corresponds to some $f(\cdot; \mathbf{u})$. A sensible approach is to search for the best importance sampling pdf $f(\cdot; \mathbf{v})$; that is, find the parameter \mathbf{v} for which the importance sampling estimator $\hat{\ell}$ has the smallest variance. Since

$$\begin{aligned} \text{Var}_{\mathbf{v}}(\hat{\ell}) &= \frac{1}{N} \text{Var}_{\mathbf{v}}(H(\mathbf{X}) W(\mathbf{X}; \mathbf{u}, \mathbf{v})) \\ &= \frac{1}{N} \left(\mathbb{E}_{\mathbf{v}} \left[H(\mathbf{X})^2 W(\mathbf{X}; \mathbf{u}, \mathbf{v})^2 \right] - \ell^2 \right), \end{aligned}$$

where $W(\mathbf{X}; \mathbf{u}, \mathbf{v}) = f(\mathbf{X}; \mathbf{u})/f(\mathbf{X}; \mathbf{v})$, the optimal \mathbf{v} is found by solving the following *variance minimization program*:

$$\min_{\mathbf{v}} \mathbb{E}_{\mathbf{v}} \left[H(\mathbf{X})^2 W(\mathbf{X}; \mathbf{u}, \mathbf{v})^2 \right].$$

This is in general not an easy problem.

2.2. Cross-entropy minimization

The idea of the CE method is to choose the importance sampling pdf g in a specified class of pdfs such that the Kullback–Leibler divergence between the optimal importance sampling pdf g^* and g is minimal. The Kullback–Leibler divergence between two pdfs g and h is given by

$$\begin{aligned}\mathcal{D}(g,h) &= \mathbb{E}_g \left[\ln \frac{g(\mathbf{X})}{h(\mathbf{X})} \right] = \int g(\mathbf{x}) \ln \frac{g(\mathbf{x})}{h(\mathbf{x})} d\mathbf{x} \\ &= \int g(\mathbf{x}) \ln g(\mathbf{x}) d\mathbf{x} - \int g(\mathbf{x}) \ln h(\mathbf{x}) d\mathbf{x}.\end{aligned}\quad (4)$$

We assume from now on that the function H is positive, that the nominal pdf f is parameterized by a finite-dimensional vector \mathbf{u} ; that is, $f(\mathbf{x}) = f(\mathbf{x}; \mathbf{u})$, and that the importance sampling pdf is $f(\cdot; \mathbf{v})$ for some parameter \mathbf{v} . The CE minimization procedure involves finding an optimal reference parameter vector, \mathbf{v}^* say, by cross-entropy minimization:

$$\begin{aligned}\mathbf{v}^* &= \operatorname{argmin}_{\mathbf{v}} \mathcal{D}(g^*, f(\cdot; \mathbf{v})) \\ &= \operatorname{argmax}_{\mathbf{v}} \int H(\mathbf{x}) f(\mathbf{x}; \mathbf{u}) \ln f(\mathbf{x}; \mathbf{v}) d\mathbf{x} \\ &= \operatorname{argmax}_{\mathbf{v}} \mathbb{E}_{\mathbf{u}} H(\mathbf{x}) \ln f(\mathbf{X}; \mathbf{v}) \\ &= \operatorname{argmax}_{\mathbf{v}} \mathbb{E}_{\mathbf{w}} H(\mathbf{x}) W(\mathbf{X}; \mathbf{u}, \mathbf{w}) \ln f(\mathbf{X}; \mathbf{v}),\end{aligned}\quad (5)$$

where \mathbf{w} is any reference parameter. This \mathbf{v}^* can be estimated via the stochastic counterpart of (5):

$$\hat{\mathbf{v}} = \operatorname{argmax}_{\mathbf{v}} \frac{1}{N} \sum_{k=1}^N H(\mathbf{X}_k) W(\mathbf{X}_k; \mathbf{u}, \mathbf{w}) \ln f(\mathbf{X}_k; \mathbf{v}), \quad (6)$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \mathbf{w})$. This leads to the following algorithm.

Algorithm 2.1 (CE Algorithm)

1. Choose a reference parameter \mathbf{w} , e.g., $\mathbf{w} = \mathbf{u}$, and generate $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \mathbf{w})$.
2. Estimate the CE optimal parameter $\hat{\mathbf{v}}$ from the stochastic program (6).
3. Generate $\mathbf{X}_1, \dots, \mathbf{X}_{N_1} \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}})$ and estimate ℓ via importance sampling, as in (3).

It can be beneficial to iterate Steps 1 and 2 a number of times in Algorithm 2.1; that is, in subsequent iterations the parameter \mathbf{w} in Step 1 is chosen as $\hat{\mathbf{v}}$, obtained in Step 2.

2.3. Updating formulae

The main benefit of using CE program (6) over the corresponding variance minimization program is that the former yields simple explicit solutions for certain important cases. The optimization of (6) is similar to the calculation of the maximum likelihood estimator of \mathbf{v} for the pdf $f(\cdot; \mathbf{v})$. Indeed, by taking the gradient of the sum in (6) and equating it to the zero vector $\mathbf{0}$, one obtains $\hat{\mathbf{v}}$ (under mild regularity conditions) as the solution to the equation

$$\frac{1}{N} \sum_{k=1}^N H(\mathbf{X}_k) W(\mathbf{X}_k; \mathbf{u}, \mathbf{w}) \nabla \ln f(\mathbf{X}_k; \mathbf{v}) = \mathbf{0}, \quad (7)$$

where $\nabla \ln f(\mathbf{X}; \mathbf{v})$ is the well-known *score function* in maximum likelihood estimation. We discuss two important special cases.

2.3.1. Exponential families

Suppose that $\{f(\cdot; \eta)\}$ forms an m -dimensional exponential family in natural parameter space; that is,

$$f(\mathbf{x}; \boldsymbol{\eta}) = c(\boldsymbol{\eta}) e^{\boldsymbol{\eta}^\top \mathbf{t}(\mathbf{x})} h(\mathbf{x}), \quad (8)$$

where $\mathbf{t}(\mathbf{x}) = (t_1(\mathbf{x}), \dots, t_m(\mathbf{x}))^\top$, $c(\boldsymbol{\eta}) > 0$, and $h(\mathbf{x}) > 0$, for all $\boldsymbol{\eta} = (v_1, \dots, v_m)^\top$ such that

$$A(\boldsymbol{\eta}) \stackrel{\text{def}}{=} -\ln c(\boldsymbol{\eta}) = \ln \int e^{\boldsymbol{\eta}^\top \mathbf{t}(\mathbf{x})} h(\mathbf{x}) d\mathbf{x} < \infty.$$

(Replace the integral by a sum in the discrete case.)

The random vector $\mathbf{t}(\mathbf{X})$ is a sufficient statistic for $\boldsymbol{\eta}$. Moreover, $\mathbb{E}_{\boldsymbol{\eta}} \mathbf{t}(\mathbf{X}) = \nabla A(\boldsymbol{\eta})$ and $\text{Cov}_{\boldsymbol{\eta}}(\mathbf{t}(\mathbf{X})) = \nabla^2 A(\boldsymbol{\eta})$. The score function becomes $\nabla \ln f(\mathbf{x}; \boldsymbol{\eta}) = \mathbf{t}(\mathbf{x}) - \nabla A(\boldsymbol{\eta})$. It follows that the solution $\hat{\boldsymbol{\eta}}$ to (7) (replacing $\mathbf{u}, \mathbf{v}, \mathbf{w}$ with $\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\omega}$) satisfies

$$\nabla A(\hat{\boldsymbol{\eta}}) = \frac{\sum_{k=1}^N H(\mathbf{X}_k) W(\mathbf{X}_k; \boldsymbol{\theta}, \boldsymbol{\omega}) \mathbf{t}(\mathbf{X}_k)}{\sum_{k=1}^N H(\mathbf{X}_k) W(\mathbf{X}_k; \boldsymbol{\theta}, \boldsymbol{\omega})}. \quad (9)$$

In particular, if the exponential family is reparameterized by the mean, $\mathbf{v} = \nabla A(\boldsymbol{\eta})$, then the CE optimal parameter $\hat{\mathbf{v}}$ is explicitly given in the right-hand side of (9).

Example 2.1 (Exponential Random Variables) Let X_1, \dots, X_m be independent and identically distributed random variables with parameter θ . We write $X_1, \dots, X_m \sim_{\text{iid}} \text{Exp}(\theta)$. Let η be the reference parameter of the importance sampling pdf $f(\mathbf{x}; \boldsymbol{\eta})$ given by

$$f(\mathbf{x}; \boldsymbol{\eta}) = \prod_{i=1}^m \eta e^{-x_i \eta} = \eta^m e^{-\eta \sum_{i=1}^m x_i},$$

which is a one-dimensional exponential family of the form (8), with $\mathbf{t}(\mathbf{x}) = -\sum_{i=1}^m x_i$ and $c(\boldsymbol{\eta}) = \eta^m$. Note that under this importance sampling pdf, $X_1, \dots, X_n \sim_{\text{iid}} \text{Exp}(\eta)$. Writing $H_k = H(\mathbf{X}_k)$ and the likelihood ratio $W_k = f(\mathbf{X}_k; \boldsymbol{\theta})/f(\mathbf{X}_k; \boldsymbol{\omega})$ in (6), the CE optimal parameter $\hat{\boldsymbol{\eta}}$ is found from

$$-\frac{d \ln c(\boldsymbol{\eta})}{d\boldsymbol{\eta}} = \frac{m}{\boldsymbol{\eta}} = \frac{\sum_{k=1}^N H_k W_k \sum_{i=1}^m X_{ki}}{\sum_{k=1}^N H_k W_k},$$

where X_{ki} is the i th component of \mathbf{X}_k . Reparameterizing the $\text{Exp}(\eta)$ distribution by the mean $v = 1/\eta$, the CE updating formula for v thus becomes:

$$\hat{v} = \frac{\sum_{k=1}^N H_k W_k \sum_{i=1}^m X_{ki}/m}{\sum_{k=1}^N H_k W_k}, \quad i = 1, \dots, n. \quad (10)$$

2.3.2. Discrete distributions

Suppose X_1, \dots, X_n are iid random variables taking values $1, \dots, m$ with probabilities v_1, \dots, v_m , respectively. Let $\mathbf{v} = (v_1, \dots, v_{m-1})$. Note that $v_m = (1 - v_1 - \dots - v_{m-1})$. The (discrete) pdf of each component X is given by

$$f(x; \mathbf{v}) = \prod_{i=1}^m v_i^{I_{\{x=i\}}} = \left(1 - \sum_{i=1}^{m-1} v_i\right)^{1 - \sum_{i=1}^{m-1} I_{\{x=i\}}} \times \prod_{i=1}^{m-1} v_i^{I_{\{x=i\}}},$$

The $m - 1$ components of the corresponding score function are given by

$$\frac{d \ln f(x; \mathbf{v})}{dv_i} = \frac{I_{\{x=i\}}}{v_i} - \frac{I_{\{x=m\}}}{v_m}, \quad i = 1, \dots, m - 1.$$

Substitution in (7), and using the abbreviations $H_k = H(X_k)$ and $W_k W(X_k; \mathbf{u}, \mathbf{w})$, shows that

$$\hat{v}_i = \frac{\sum_{k=1}^N H_k W_k I_{\{X_k=i\}}}{\sum_{k=1}^N H_k W_k I_{\{X_k=m\}}} \hat{v}_m.$$

The parameter \hat{v}_m can be found from the fact that $\hat{v}_1 + \dots + \hat{v}_m = 1$. It follows that

$$\hat{v}_i = \frac{\sum_{k=1}^N H_k W_k I_{\{X_k=i\}}}{\sum_{k=1}^N H_k W_k}, \quad i = 1, \dots, m.$$

2.4. Rare-event simulation

Often the quantity of interest $\ell = \mathbb{E}H(\mathbf{X})$ is a probability of the form $\ell = \mathbb{P}(S(\mathbf{X}) \geq \gamma)$ for some function S and *level* γ ; that is, $H(\mathbf{x}) = I_{\{S(\mathbf{X}) \geq \gamma\}}$. A complication in solving (6) or (7) occurs when ℓ is a rare-event probability. In that case the optimization program and the updating formula becomes useless for moderate sample size N , because all (or most) of the values $H(\mathbf{X}_k)$ are zero. One remedy is to use a *multi-level* CE procedure instead, where a sequence of reference parameters and levels is constructed with the goal that the former converges to \mathbf{v}^* and the latter to γ . The idea is to first choose a level parameter $\hat{\ell}_1$ for which the event $S(\mathbf{X}) \geq \hat{\gamma}_1$ is not too rare under the original pdf $f(\cdot; \mathbf{u})$, and then estimate the CE optimal parameter, say \hat{v}_1 for this level via (6) (which is now not devoid of meaning). Specifically, $\hat{\gamma}_1$ is chosen as the sample $(1 - \varrho)$ -quantile of $S(\mathbf{X})$, based on a random sample $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{iid} f(\cdot; \mathbf{u})$. This procedure is then iterated by choosing $\hat{\gamma}_2$ as the sample $(1 - \varrho)$ -quantile of $S(\mathbf{X})$, based on a random sample from $f(\cdot; \hat{v}_1)$, estimating the optimal CE parameter \hat{v}_2 from (6), and so on. The complete procedure is summarized as follows; see, e.g., Rubinstein and Kroese (2007, Page 238).

Algorithm 2.2 (CE Algorithm for Rare-Event Estimation)

1. Define $\hat{\mathbf{v}}_0 = \mathbf{u}$. Let $N^e = \lceil \varrho N \rceil$. Set $t = 1$ (iteration counter).
2. Generate $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{iid} f(\cdot; \hat{\mathbf{v}}_{t-1})$. Calculate $S_i = S(\mathbf{X}_i)$ for all i , and order these from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t$ be the sample $(1 - \varrho)$ -quantile of performances; that is, $\hat{\gamma}_t = S_{(N-N^e+1)}$. If $\hat{\gamma}_t > \gamma$, reset $\hat{\gamma}_t$ to γ .

3. Use the **same** sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ to solve the stochastic program (6), with $\mathbf{w} = \hat{\mathbf{v}}_{t-1}$. Denote the solution by $\hat{\mathbf{v}}_t$.
4. If $\hat{\gamma}_t < \gamma$, set $t = t + 1$ and reiterate from Step 2; otherwise, proceed with Step 5.
5. Let T be the final iteration counter. Generate $\mathbf{X}_1, \dots, \mathbf{X}_{N_1} \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}}_T)$ and estimate ℓ via importance sampling, as in (3).

Apart from specifying the family of sampling pdfs, the sample sizes N and N_1 , and the rarity parameter ϱ (typically between 0.01 and 0.1), the algorithm is completely self-tuning. The sample size N for determining a good reference parameter can usually be chosen much smaller than the sample size N_1 for the final importance sampling estimation, say $N = 1000$ versus $N_1 = 100,000$. Under certain technical conditions the deterministic version of Algorithm 2.2 is guaranteed to terminate (reach level γ) provided that ϱ is chosen small enough; see Section 3.5 of [Rubinstein and Kroese \(2004\)](#).

Example 2.2 (Stochastic Activity Network) Figure 1 shows an example of a *stochastic activity network*. Such networks are frequently used in process management to schedule concurrent activities of some project from start to finish. Each arc in the network corresponds to an activity, and is weighted by the duration of that activity. The nodes in the network represent milestones. Any activity originating from a milestone can only be started once all activities leading to that milestone have been completed. The maximal project duration corresponds to the length of the longest path in the graph. Figure 1 shows a stochastic activity network with 10 activities. Suppose the durations of the activities are independent exponential random variables X_1, \dots, X_{10} , each with means 1.

Let $S(\mathbf{X})$ denote length of the longest path in the graph; that is,

$$S(\mathbf{X}) = \max_i L_i,$$

where

$$\begin{aligned} L_1 &= X_1 + X_4 + X_9, \\ L_2 &= X_3 + X_6 + X_9, \\ L_3 &= X_3 + X_8, \\ L_4 &= X_3 + X_7 + X_{10}, \\ L_5 &= X_2 + X_5 + X_{10}. \end{aligned}$$

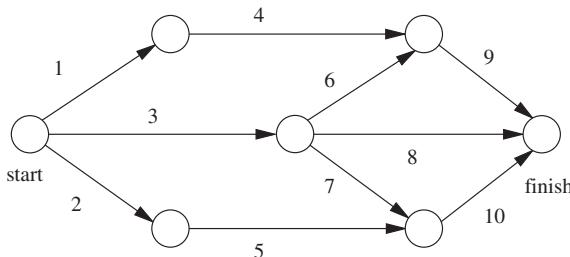


Fig. 1. A stochastic activity network.

Table 1
Convergence of the sequence $\{(\hat{\gamma}_t, \hat{\mathbf{v}}_t)\}$

t	$\hat{\gamma}_t$	$\hat{\mathbf{v}}_t$									
0	–	1	1	1	1	1	1	1	1	1	1
1	7.05	1.479	1.453	1.763	1.473	1.441	1.398	1.427	1.147	1.707	1.732
2	11.09	1.794	1.826	2.422	1.773	1.798	1.735	1.731	1.176	2.449	2.444
3	14.69	2.034	2.034	3.056	2.213	2.167	2.125	1.982	1.040	3.070	2.977
4	17.87	2.097	2.297	3.570	2.352	2.327	2.281	2.593	1.021	3.614	3.842
5	20.00	2.936	2.548	3.683	2.537	2.628	2.454	2.227	1.209	4.004	3.683

Suppose the objective is to estimate the rare-event probability $\mathbb{P}(S(\mathbf{X}) \geq 20)$ using importance sampling where the random vector $\mathbf{X} = (X_1, \dots, X_{10})$ has independent exponentially distributed components with mean vector $\mathbf{v} = (v_1, \dots, v_{10})$. Note that the nominal pdf is obtained by setting $v_i = 1$ for all i . At the t th iteration of the multi-level CE Algorithm 2.2, the solution to (6) with $H(\mathbf{X}) = \mathbf{I}_{\{S(\mathbf{X}) \geq \hat{\gamma}_t\}}$ is, similar to (10), given by

$$\hat{v}_{t,i} = \frac{\sum_{k=1}^N \mathbf{I}_{\{S(\mathbf{X}_k) \geq \hat{\gamma}_t\}} W_k X_{ki}}{\sum_{k=1}^N \mathbf{I}_{\{S(\mathbf{X}_k) \geq \hat{\gamma}_t\}} W_k}, \quad (11)$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}}_{t-1})$, $W_k = f(\mathbf{X}_k; \mathbf{u})/f(\mathbf{X}_k; \hat{\mathbf{v}}_{t-1})$, and X_{ki} is the i th element of \mathbf{X}_k .

Table 1 lists the successive estimates for the optimal importance sampling parameters obtained from the multi-level CE algorithm, using $N = 10^5$ and $\varrho = 0.1$.

The last step in Algorithm 2.2 gives an estimate of $1.72 \cdot 10^{-6}$ with an estimated relative error of 2%, using a sample size of $N_1 = 10^6$. A typical crude Monte Carlo estimate (that is, taking $\mathbf{v} = \mathbf{u} = (1, 1, \dots, 1)$) using the same sample size is $3 \cdot 10^{-6}$, with an estimated relative error of 60%, and is therefore of little use.

3. Extensions

3.1. Sampling directly from the zero-variance distribution

For certain rare-event estimation problems it is possible to (approximately) sample directly from the zero-variance importance sampling pdf g^* , for example via Markov chain Monte Carlo. By sampling directly from g^* one can estimate the CE optimal parameters in a single step and without likelihood ratios. The idea was first introduced in Chan (2010). This approach could be advantageous for high-dimensional problems, where the *likelihood degeneracy* can pose a serious impediment to importance sampling; see, for example, Rubinstein and Kroese (2007, Page 133).

To explain the idea, suppose $\ell = \mathbb{P}(S(\mathbf{X}) \geq \gamma)$ is the rare event of interest, where $\mathbf{X} \sim f(\cdot; \mathbf{u})$. The zero-variance importance sampling density g^* is simply the conditional pdf f given the event $S(\mathbf{X}) \geq \gamma$; that is,

$$g^*(\mathbf{x}) = \frac{f(\mathbf{x}; \mathbf{u}) \mathbf{I}_{\{S(\mathbf{x}) \geq \gamma\}}}{\ell}.$$

The CE optimal parameter \mathbf{v}^* is

$$\begin{aligned}\mathbf{v}^* &= \operatorname{argmax}_{\mathbf{v}} \int I_{\{S(\mathbf{x}) \geq \gamma\}} f(\mathbf{x}; \mathbf{u}) \ln f(\mathbf{x}; \mathbf{v}) d\mathbf{x} \\ &= \operatorname{argmax}_{\mathbf{v}} \mathbb{E}_{g^*} \ln f(\mathbf{x}; \mathbf{v}) d\mathbf{x},\end{aligned}$$

which can be estimated via the sample average approximation:

$$\hat{\mathbf{v}}^* = \operatorname{argmax}_{\mathbf{v}} \frac{1}{N} \sum_{k=1}^N \ln f(\mathbf{X}_k; \mathbf{v}), \quad (12)$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N$ is a (approximate) sample from g^* . This leads to the following algorithm; see [Chan and Kroese \(2011\)](#).

Algorithm 3.1 (CE Algorithm via the Zero-Variance Distribution)

1. Generate a random sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ from the density g^* and find the solution to (12).
2. Generate a sample $\mathbf{X}_1, \dots, \mathbf{X}_M$ from the density $f(\cdot; \hat{\mathbf{v}}^*)$ and estimate ℓ via importance sampling, as in (3).

Note that no likelihood ratio or indicator is involved. As a result, the algorithm does not only afford substantial computational saving in high-dimensional settings, its solution is more robust and numerically stable as well. Generating draws from g^* , however, requires additional effort, but with the advent of Markov chain Monte Carlo (MCMC) methods, this problem is well studied and a variety of techniques are available to our disposal. In particular, sampling from the zero-variance pdf g^* can be achieved without knowledge of ℓ . The number of draws required to estimate $\hat{\mathbf{v}}^*$ is typically much smaller than that required in the multi-level CE algorithm. As (12) is a maximum likelihood type estimator, where sample is taken from g^* instead of $f(\cdot; \mathbf{v})$, the solution can often be obtained analytically. Note, finally, that in this approach the function S must be explicitly available, while in the standard CE method any importance sampling pdf can be used, in conjunction with the likelihood ratio.

Example 3.1 (Binomial Distribution) To compare the quality of the optimal reference parameter estimators for the multi-level CE Algorithm 2.2 and the zero-variance CE Algorithm 3.1, consider the estimation of $\mathbb{P}(S(\mathbf{X}) \geq \gamma)$, where $S(\mathbf{X}) = X_1 + \dots + X_n$, $\gamma = n\beta$ for some $\beta \in (0, 1)$, and $X_i \sim \text{Ber}(p_i)$, $i = 1, \dots, n$, independently. The nominal density is thus $f(\mathbf{x}; \mathbf{p}) = \prod_{i=1}^n p_i^{x_i} (1-p_i)^{1-x_i}$, where $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{p} = (p_1, \dots, p_n)$. We wish to locate the optimal importance density within the parametric family $f(\mathbf{x}; \mathbf{q})$ indexed by $\mathbf{q} = (q_1, \dots, q_n)$, where $q_i \in (0, 1)$ for $i = 1, \dots, n$. The CE optimal parameter \mathbf{q}^* follows from the maximization program

$$\max_{\mathbf{q}} \sum_{\mathbf{x}: S_n(\mathbf{x}) \geq \gamma} \left(\prod_{i=1}^n p_i^{x_i} (1-p_i)^{1-x_i} \right) \left(\sum_{i=1}^n x_i \ln q_i + (1-x_i) \ln (1-q_i) \right),$$

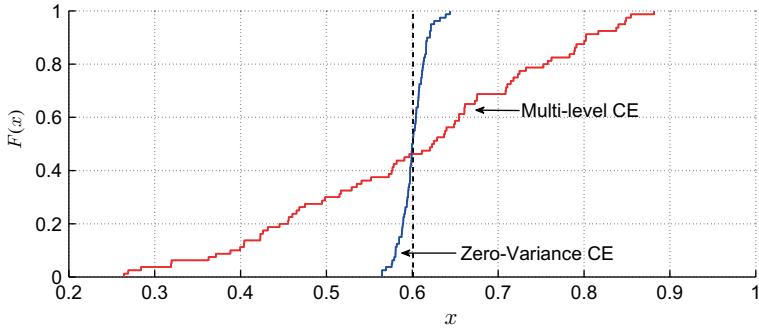


Fig. 2. The empirical distribution function of the CE estimates.

which yields the closed-form expression

$$q_j^* = \frac{\sum_{\mathbf{x}: S_n(\mathbf{x}) \geq \gamma} x_j \prod_{i=1}^n p_i^{x_i} (1-p_i)^{1-x_i}}{\sum_{\mathbf{x}: S_n(\mathbf{x}) \geq \gamma} \prod_{i=1}^n p_i^{x_i} (1-p_i)^{1-x_i}}, \quad j = 1, \dots, n.$$

In particular, if all the p_i are identical, then $q_j^* = \lceil \beta n \rceil / n$. We estimate \mathbf{q}^* via the multi-level CE procedure and by sampling from the zero-variance pdf. As a numerical example, we first set $n = 80$, $\beta = 0.6$, and $p_1 = \dots = p_n = 0.1$. For the multi-level CE method, we set $N = 10,000$ and $\rho = 0.01$. The algorithm terminates at the fifth iteration, requiring a total of 50,000 draws. For the zero-variance CE procedure, we run a Gibbs sampler with 10 parallel chains, each has a length of 1000, and the total budget is therefore 10,000. It is also worth mentioning that drawing from g^* via the Gibbs sampler in this case only requires generating Bernoulli draws. The empirical cumulative distribution functions (cdf) of the CE and “zero-variance CE” estimates, together with the optimal reference parameter calculated analytically, are presented in Fig. 2.

It is evident that the multi-level CE estimates fluctuate more widely compared to those obtained by the zero-variance version, even though the simulation budget for the former is five times as large. Using a sample size of $N = 50,000$, typical importance sampling estimates for the multi-level and zero-variance approaches are $3 \cdot 10^{-28}$ and $7.96 \cdot 10^{-28}$, respectively, with estimated relative errors 0.3 and 0.01. The true probability is approximately $8.10 \cdot 10^{-28}$. It is important to note that the multi-level CE estimation procedure can be considerably improved by simply increasing the sample size for estimating the parameters, for example from 10,000 to 100,000. Similar experiments indicate that as the dimension n of the problem gets larger, the multi-level CE estimates become more unreliable, compared to the zero-variance counterpart, which are essentially unaffected by the increase in dimension.

The result from this toy example suggests a reason why the multi-level CE method fails to give accurate estimates in high-dimensional settings: the reference parameter vector obtained is suboptimal, and therefore the resulting importance density does not sufficiently mimic the behavior of g^* . In principle one can increase the accuracy of the multi-level CE estimates by increasing the sample size N or the rarity parameter ϱ . In either case, however, the total simulation effort would increase, and

in moderately high-dimensional problems, this approach might not be practical. On the other hand, the result also suggests that if we avoid the multi-level maximization procedure and estimate \mathbf{v}^* directly via (12), we can improve the performance of the standard CE procedure.

Example 3.2 (Stochastic Activity Network (Continued)) This is to show how the zero-variance CE approach can be applied to the stochastic activity network of Example 2.2.

We can sample from the zero-variance pdf g^* using the Gibbs sampler. This requires sampling from the marginal distributions of g^* conditional on the values of the other components. Consider first sampling X_1 conditional on x_2, \dots, x_{10} . If any of the lengths $L_i, i = 2, \dots, 5$ is greater than or equal to γ , then the marginal distribution of X_1 given x_2, \dots, x_{10} is simply equal to the nominal distribution (that is, $\text{Exp}(1)$). However, if every L_2, \dots, L_{10} is less than γ , then in order for L_1 to exceed γ , X_1 has to be greater than or equal to $\min\{\gamma - x_4 - x_6, 0\}$. Hence, to sample from the marginal conditional distribution in this case, draw $Z \sim \text{Exp}(1)$ and compute $X_1 = Z + \min\{\gamma - x_4 - x_6, 0\}$. That is X_1 is drawn from a truncated exponential distribution. Similarly, given $x_1, x_3, \dots, x_{10}, X_2$ is drawn from its original distribution if $\max_{i \leq 4} L_i \geq \gamma$; otherwise, set $X_2 = \min\{\gamma - x_5 - x_{10}, 0\} + Z$, where $Z \sim \text{Exp}(1)$. More precisely, the Gibbs sampling procedure for this problem is as follows.

Algorithm 3.2 (Gibbs Sampling for the Stochastic Activity Network)

1. Initialize \mathbf{X} such that $S(\mathbf{X}) \geq \gamma$. Set $t = 1$ (counter).
2. For $i = 1$ to $n = 10$:
 - a. Let \mathcal{P}_i be the set of paths containing link i .
 - b. If $\max_{k \notin \mathcal{P}_i} L_k \geq \gamma$, then draw $X_i \sim \text{Exp}(1)$.
 - c. Otherwise, set $X_i = 0$ and compute $L = \max_{k \in \mathcal{P}_i} L_k$. Draw $Z \sim \text{Exp}(1)$ and set $X_i = Z + \max\{\gamma - L, 0\}$.
3. Set $t = t + 1$. If $t > N$ (sample size) stop; otherwise, return to Step 2.

The estimate and relative error are comparable to the ones obtained via the multi-level approach. See also [Chan \(2010\)](#) for an application to a high-dimensional network.

3.2. Transformed likelihood ratio

The *transform likelihood ratio* (TLR) method ([Kroese and Rubinstein, 2004](#)) is a convenient constructing efficient importance sampling estimators.

The idea is to apply a simple *change of variable* step to the estimation problem and then to apply the CE method to find the optimal importance sampling parameter for the transformed problem. Suppose the objective is to estimate $\ell = \mathbb{E}H(\mathbf{X})$. The main step is to write \mathbf{X} as a function of another random vector, say as

$$\mathbf{X} = G(\mathbf{Z}). \quad (13)$$

If we define

$$\tilde{H}(\mathbf{Z}) = H(G(\mathbf{Z})),$$

then estimating ℓ is equivalent to estimating

$$\ell = \mathbb{E}[\tilde{H}(\mathbf{Z})]. \quad (14)$$

As an example, consider the one-dimensional case where $X \sim \text{Weib}(\alpha, \lambda)$. Since X has the same distribution as $Z^{1/\alpha}/\lambda$, where $Z \sim \text{Exp}(1)$, we have $\tilde{H}(Z) = H(\lambda^{-1}Z^{1/\alpha})$ and $\ell = \mathbb{E}[H(\lambda^{-1}Z^{1/\alpha})]$.

To apply the cross-entropy method, assume that \mathbf{Z} has a density $h(\mathbf{z}; \boldsymbol{\theta})$ in some class of densities $\{h(\mathbf{z}; \boldsymbol{\eta})\}$. Then we can seek to estimate ℓ efficiently via importance sampling. In particular, by analogy to (3), we obtain the following TLR estimator:

$$\hat{\ell} = \frac{1}{N} \sum_{k=1}^N \tilde{H}(\mathbf{Z}_k) \tilde{W}(\mathbf{Z}_k; \boldsymbol{\theta}, \boldsymbol{\eta}), \quad (15)$$

where

$$\tilde{W}(\mathbf{Z}_k; \boldsymbol{\theta}, \boldsymbol{\eta}) = \frac{h(\mathbf{Z}_k; \boldsymbol{\theta})}{h(\mathbf{Z}_k; \boldsymbol{\eta})}$$

and $\mathbf{Z}_k \sim h(\mathbf{z}; \boldsymbol{\eta})$. As an example, consider again the $\text{Weib}(\alpha, \lambda)$ case. Using the transform $X = Z^{1/\alpha}/\lambda$, we could apply importance sampling to $Z \sim \text{Exp}(1)$, using an $\text{Exp}(\eta)$ class of distributions. Thus, $h(z; \eta) = \eta e^{-\eta z}$ is the importance sampling pdf, with $\eta = \theta = 1$ as the nominal parameter. Hence, in this case, $\hat{\ell}$ in (15) reduces to

$$\hat{\ell} = \frac{1}{N} \sum_{k=1}^N \tilde{H}\left(\lambda^{-1}Z_k^{1/\alpha}\right) \tilde{W}(Z_k; \theta, \eta), \quad (16)$$

with

$$\tilde{W}(Z_k; \theta, \eta) = \frac{h(Z_k; \theta)}{h(Z_k; \eta)} = \frac{\theta e^{-\theta Z_k}}{\eta e^{-\eta Z_k}}$$

and $Z_k \sim \text{Exp}(\eta)$.

To find the optimal parameter vector $\boldsymbol{\eta}^*$ of the TLR estimator (15) we can solve, by analogy to (5), the following CE program:

$$\boldsymbol{\eta}^* = \underset{\boldsymbol{\eta}}{\operatorname{argmax}} \mathbb{E}_{\boldsymbol{\tau}} [\tilde{H}(\mathbf{Z}) \tilde{W}(\mathbf{Z}; \boldsymbol{\theta}, \boldsymbol{\tau}) \ln h(\mathbf{Z}; \boldsymbol{\eta})] \quad (17)$$

and similarly for the stochastic counterpart of (17).

To obtain simple updating formulas one would typically choose the distribution of \mathbf{Z} from an exponential family of distributions, as is explained in Section 2.3. Below we present the TLR algorithm for estimating $\ell = \mathbb{E}_f[H(\mathbf{X})]$, assuming that \mathbf{X} is a random vector with independent, continuously distributed components.

Algorithm 3.3 (TLR Method)

1. For a given random vector \mathbf{X} , find a transformation G such that $\mathbf{X} = G(\mathbf{Z})$, with $\mathbf{Z} \sim h(\mathbf{z}; \boldsymbol{\theta})$. For example, take \mathbf{Z} with all components being iid and distributed according to an exponential family (e.g., $\text{Exp}(1)$).
2. Generate a random sample $\mathbf{Z}_1, \dots, \mathbf{Z}_N$ from $h(\cdot; \boldsymbol{\tau})$.
3. Solve the stochastic counterpart of the program (17). Iterate if necessary. Denote the solution by $\hat{\boldsymbol{\eta}}$.

4. Generate a (larger) random sample $\mathbf{Z}_1, \dots, \mathbf{Z}_{N_1}$ from $h(\cdot; \hat{\eta})$ and estimate $\ell = \mathbb{E}[H(G(\mathbf{Z}))]$ via the TLR estimator (15), taking $\eta = \hat{\eta}$.

The advantage of the TLR method is its universality and its ability to avoid the computational burden while directly delivering the analytical solution of the stochastic counterpart of the program (17).

Example 3.3 (Elliptical Distributions) A random vector $\mathbf{X} = (X_1, \dots, X_n)$ is said to have an *elliptical distribution* with location vector μ , dispersion matrix Σ , and radial cdf F , written $\mathbf{X} \sim \text{Ellipt}(\mu, \Sigma, F)$, if \mathbf{X} can be written in the form

$$\mathbf{X} = \mu + R\mathbf{B}\mathbf{Y}, \quad (18)$$

where \mathbf{Y} is a uniform vector on the n -dimensional sphere, R is a random variable with cdf F , independent of \mathbf{Y} , and B is a matrix such that $BB^\top = \Sigma$. Note that a random vector that is uniformly distributed on the n -dimensional sphere can be obtained by normalizing an n -dimensional standard normal random vector: $\mathbf{Y} = \mathbf{Z}/\|\mathbf{Z}\|$, with $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, I_n)$; see, for example, Kroese et al. (2011); Blanchet and Rojas-Nandayapa (in press) consider efficient simulation procedures for estimating probabilities of the form

$$\ell = \mathbb{P}(e^{X_1} + \dots + e^{X_n} \geq \gamma), \quad (19)$$

where $\mathbf{X} \sim \text{Ellipt}(\mu, \Sigma, F)$. We show how such quantities can be quickly computed via the TLR method. Defining

$$S(R, \mathbf{Y}) = e^{X_1} + \dots + e^{X_n},$$

where the $\{X_i\}$ are related to the $\{Y_i\}$ and R via (18), the idea is to write (19) as

$$\ell = \mathbb{P}(S(R, \mathbf{Y}) \geq \gamma)$$

and to estimate the latter by performing importance sampling on the distribution of R only. Suppose, for simplicity, that $R \sim \text{Exp}(1)$ and that the importance sampling distribution is $\text{Exp}(1/\hat{v}_T)$ where \hat{v}_T is obtained at the last step of a CE procedure. This means that ℓ is estimated as

$$\hat{\ell} = \frac{1}{N_1} \sum_{k=1}^{N_1} I_{\{S(R_k, \mathbf{Y}_k) \geq \gamma\}} W_k, \quad (20)$$

where $W_k = \hat{v}_T e^{-(1-1/\hat{v}_T)R_k}$. Because $\{\text{Exp}(1/v), v > 0\}$ is an exponential family parameterized by its mean, the optimal reference parameter at the t -th iteration of the CE algorithm is given by

$$\hat{v}_t = \frac{\sum_{k=1}^N I_{\{S(R_k, \mathbf{Y}_k) \geq \hat{v}_t\}} W_k R_{ki}}{\sum_{k=1}^N I_{\{S(R_k, \mathbf{Y}_k) \geq \hat{v}_t\}} W_k}, \quad (21)$$

where $W_k = \hat{v}_{t-1} e^{-(1-1/\hat{v}_{t-1})R_k}$. This leads to the following procedure.

Algorithm 3.4

1. Set $\hat{v}_0 = u = 1$ and $t = 1$.
2. Generate $R_1, \dots, R_N \sim_{\text{iid}} \text{Exp}(\hat{v}_{t-1})$ and (independently) $\mathbf{Z}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_n), k = 1, \dots, N$. Let $\mathbf{Y}_k = \mathbf{Z}_k / \|\mathbf{Z}_k\|, k = 1, \dots, N$.
3. Set $\mathbf{X}_k = \boldsymbol{\mu} + R_k B \mathbf{Y}_k$, calculate the performances $S(R_k, \mathbf{Y}_k)$ and order these from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t$ be the sample $(1 - \varrho)$ -quantile of performances. If $\hat{\gamma}_t > \gamma$, reset $\hat{\gamma}_t$ to γ .
4. Update \hat{v}_t via (21), using the **same** R_1, \dots, R_N as obtained in Step 2.
5. If $\hat{\gamma}_t < \gamma$, set $t = t + 1$ and reiterate from Step 2; otherwise, proceed with Step 6.
6. Let T be the final iteration counter. Generate $R_1, \dots, R_{N_1} \sim_{\text{iid}} \text{Exp}(\hat{v}_T)$ and $\mathbf{Y}_1, \dots, \mathbf{Y}_{N_1}$, and estimate ℓ via importance sampling as in (20).

As a numerical example, consider the 10-dimensional case with $\boldsymbol{\mu} = \mathbf{0}, \Sigma = I_{10}$ (identity), $R \sim \text{Exp}(1)$, and $\gamma = 3 \cdot 10^3$. Using a sample size of $N = 10^3$ the following CE parameter was found after three iterations: $\hat{v}_3 = 12.05$. Using importance sampling with a sample size of 10^6 the rare-event probability was estimated as $3.91 \cdot 10^{-6}$ with an estimated relative error of 1.2%. For $\gamma = 3 \cdot 10^5$, using the same sample sizes, the probability was estimated (after five iterations) as $9.3 \cdot 10^{-9}$ with an estimated relative error of 2.6%.

3.3. Root finding

In many applications one needs to estimate, for given ℓ , the *root*, γ , of the nonlinear equation

$$\mathbb{P}_{\mathbf{u}}(S(\mathbf{X}) \geq \gamma) = \mathbb{E}_{\mathbf{u}}[I_{\{S(\mathbf{X}) \geq \gamma\}}] = \ell \quad (22)$$

rather than estimate ℓ itself. We call such a problem a *root-finding* problem.

One can obtain γ using the CE method, by finding a good reference vector $\hat{\mathbf{v}}_T$ such that γ can be estimated accurately as the smallest number $\hat{\gamma}$ such that

$$\frac{1}{N_1} \sum_{k=1}^{N_1} I_{\{S(\mathbf{X}_k) \geq \hat{\gamma}\}} W(\mathbf{X}_k; \mathbf{u}, \hat{\mathbf{v}}_T) \leq \ell. \quad (23)$$

To find such a reference vector $\hat{\mathbf{v}}_T$ one can simply modify the multi-level CE Algorithm 2.2 as follows.

Algorithm 3.5 (Root-Finding Algorithm)

1. Define $\hat{\mathbf{v}}_0 = \mathbf{u}, N^e = \lceil (1 - \varrho)N \rceil$. Set $t = 1$.
2. Generate a random sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ from the density $f(\cdot; \hat{\mathbf{v}}_{t-1})$.
3. Calculate the performances $S(\mathbf{X}_1), \dots, S(\mathbf{X}_N)$. Order the performances from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t = S_{(N^e)}$.
4. Calculate $\hat{\ell}_t = \max\{\ell, \frac{1}{N} \sum_{k=1}^N I_{\{S(\mathbf{X}_k) \geq \hat{\gamma}_t\}} W(\mathbf{X}_k; \mathbf{u}, \hat{\mathbf{v}}_{t-1})\}$.
5. Solve the stochastic program (6) with $\mathbf{w} = \hat{\mathbf{v}}_{t-1}$, using the **same** sample $\mathbf{X}_1, \dots, \mathbf{X}_N$. Denote the solution by $\hat{\mathbf{v}}_t$.
6. If $\hat{\ell}_t = \ell$, proceed to Step 7; otherwise, let $t = t + 1$ and reiterate from Step 2.

Table 2
Convergence of the sequence $\{\hat{\ell}_t, \hat{v}_t\}$

t	$\hat{\ell}_t$	\hat{v}_t									
0	—	1	1	1	1	1	1	1	1	1	1
1	0.1	1.46	1.44	1.77	1.44	1.44	1.44	1.43	1.16	1.72	1.70
2	$4.28 \cdot 10^{-3}$	1.74	1.83	2.54	1.79	1.81	1.75	1.78	1.19	2.39	2.45
3	$1.75 \cdot 10^{-4}$	2.04	1.99	3.25	2.10	2.01	2.00	2.24	1.07	3.11	3.11
4	$1.00 \cdot 10^{-5}$	2.42	2.28	3.92	2.18	2.28	2.15	2.68	1.03	3.54	4.12

7. Estimate γ via the right-hand side of (23), using a sample $\mathbf{X}_1, \dots, \mathbf{X}_{N_1} \sim f(\cdot; \hat{v}_T)$, where T is the final iteration number.

Example 3.4 (Root Finding for the Stochastic Activity Network) Consider the stochastic activity network in Example 2.2. Suppose we wish to estimate for which γ

$$\mathbb{P}(S(\mathbf{X}) \geq \gamma) = 10^{-5}.$$

Table 2 shows a typical outcome of Algorithm 3.5, with a sample size of $N = 10^5$ and $\varrho = 0.1$. A typical estimate of γ using a sample size of $N_1 = 10^6$ is 18.08 with an estimated relative error of 0.1%.

Acknowledgment

This work was supported by the Australian Research Council under Grant No. DP0985177.

References

- Asmussen, S., Kroese, D.P., Rubinstein, R.Y., 2005. Heavy tails, importance sampling and cross-entropy. Stoch. Models 21(1), 57–76.
- Blanchet, J.H., Rojas-Nandayapa, L., 2011. Efficient simulation of tail probabilities of sums of dependent random variables. J. Appl. Probab. 48A, 147–164.
- Chan, J.C.C., 2010. Advanced Monte Carlo methods with applications in finance. PhD Thesis. University of Queensland.
- Chan, J.C.C., Kroese, D.P., 2010. Efficient estimation of large portfolio loss probabilities in t-copula models. Eur. J. Oper. Res. 2050 (2), 361–367.
- Chan, J.C.C., Kroese, D.P., 2011. Rare-event probability estimation with conditional Monte Carlo. Ann. Oper. Res. 189 (1), 155–165.
- Chan, J.C.C., Kroese, D.P., 2012. Improved cross-entropy method for estimation. Stat. Comput. 22 (5), 1031–1040.
- Chan, J.C.C., Glynn P.W., Kroese, D.P., 2011. A comparison of cross-entropy and variance minimization strategies. J. Appl. Prob. 48A, 183–194.
- de Boer, P.T., 2000. Analysis and efficient simulation of queueing models of telecommunication systems. PhD Thesis. University of Twente.
- de Boer, P.T., Kroese, D.P., Rubinstein, R.Y., 2004. A fast cross-entropy method for estimating buffer overflows in queueing networks. Manag. Sci. 500 (7), 883–895.

- de Boer, P.T., Kroese, D.P., Mannor, S., Rubinstein, R.Y., 2005. A tutorial on the cross-entropy method. *Ann. Oper. Res.* 1340 (1), 19–67.
- Evans, G.E., 2009. Parallel and sequential Monte Carlo methods with applications. PhD Thesis. The University of Queensland, Brisbane.
- Evans, G.E., Keith, J.M., Kroese, D.P., 2007. Parallel cross-entropy optimization. In: Proceedings of the 2007 Winter Simulation Conference, Washington, DC, pp. 2196–2202.
- Homem-de-Mello, T., 2007. A study on the cross-entropy method for rare event probability estimation. *INFORMS J. Comput.* 190 (3), 381–394.
- Hui, K.-P., Bean, N., Kraetzl, M., Kroese, D.P., 2005. The cross-entropy method for network reliability estimation. *Ann. Oper. Res.* 134, 101–118.
- Kroese, D.P., Hui, K.-P., 2006. In: Computational Intelligence in Reliability Engineering, chapter 3: Applications of the Cross-Entropy Method in Reliability. Springer-Verlag, New York.
- Kroese, D.P., Rubinstein, R.Y., 2004. The transform likelihood ratio method for rare event simulation with heavy tails. *Queueing Syst.* 46, 317–351.
- Kroese, D.P., Taimre, T., Botev, Z.I., 2011. Handbook of Monte Carlo Methods. John Wiley & Sons, New York.
- Rao, C.R., 2010. Entropy and cross entropy: characterizations and applications. In: Alladi, K., Klauder, J., Rao, C.R. (Eds.), *The Legacy of Alladi Ramakrishnan in the Mathematical Sciences*. Springer-Verlag, New York, pp. 359–367.
- Ridder, A., 2005. Importance sampling simulations of Markovian reliability systems using cross-entropy. *Ann. Oper. Res.* 1340 (1), 119–136.
- Rubinstein, R.Y., 1997. Optimization of computer simulation models with rare events. *Eur. J. Oper. Res.* 990 (1), 89–112.
- Rubinstein, R.Y., 1999. The cross-entropy method for combinatorial and continuous optimization. *Methodol. Comput. Appl. Probab.* 10 (2), 127–190.
- Rubinstein, R.Y., 2001. Combinatorial optimization, cross-entropy, ants and rare events. In: Uryasev, S., Pardalos, P.M. (Eds.), *Stochastic Optimization: Algorithms and Applications*, Kluwer, Dordrecht, pp. 304–358.
- Rubinstein, R.Y., Kroese, D.P., 2004. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization*. Monte Carlo Simulation and Machine Learning. Springer-Verlag, New York.
- Rubinstein, R.Y., Kroese, D.P., 2007. *Simulation and the Monte Carlo Method*, second ed. John Wiley & Sons, New York.
- Taimre, T., 2009. Advances in cross-entropy methods. PhD Thesis. The University of Queensland, Brisbane.

The Cross-Entropy Method for Optimization

*Zdravko I. Botev¹, Dirk P. Kroese², Reuven Y. Rubinstein³,
and Pierre L'Ecuyer¹*

¹*Department of Computer Science and Operations Research, Université de Montréal, Montréal, Québec, Canada H3C 3J7*

²*School of Mathematics and Physics, The University of Queensland, Brisbane 4072, Australia*

³*Faculty of Industrial Engineering and Management, Technion, Haifa, Israel*

Abstract

The cross-entropy method is a versatile heuristic tool for solving difficult estimation and optimization problems, based on Kullback–Leibler (or cross-entropy) minimization. As an optimization method it unifies many existing population-based optimization heuristics. In this chapter we show how the cross-entropy method can be applied to a diverse range of combinatorial, continuous, and noisy optimization problems.

Keywords: cross-entropy, continuous and combinatorial optimization, noisy optimization, population Monte Carlo, network planning problem, likelihood maximization

1. Introduction

The *cross-entropy (CE) method* was proposed by Rubinstein (1997) as an adaptive importance sampling procedure for the estimation of rare-event probabilities that uses the *cross-entropy* or *Kullback–Leibler divergence* as a measure of closeness between two sampling distributions. Subsequent work by Rubinstein (1999, 2001) has shown that many optimization problems can be translated into a rare-event estimation problem. As a result, adaptive importance sampling methods such as the CE method can be utilized as randomized algorithms for optimization. The gist of the idea is that the probability of locating an optimal or near-optimal solution using naive random search is a rare-event probability. The cross-entropy method can be used to gradually change the sampling distribution of the random search so that the rare event is more likely to occur. For this purpose, using the CE distance, the method

estimates a sequence of sampling distributions that converges to a distribution with probability mass concentrated in a region of near-optimal solutions.

To date, the CE method has been successfully applied to mixed integer nonlinear programming (Kothari and Kroese, 2009); continuous optimal control problems (Sani, 2009; Sani and Kroese, 2008); continuous multi-extremal optimization (Kroese et al., 2006); multidimensional independent component analysis (Szabó et al., 2006); optimal policy search (Busoniu et al., 2010); clustering (Botev and Kroese, 2004; Kroese et al., 2007b; Boubezoula et al., 2008); signal detection (Liu et al., 2004); DNA sequence alignment (Keith and Kroese, 2002; Pihur et al., 2007); noisy optimization problems such as optimal buffer allocation (Alon et al., 2005); resource allocation in stochastic systems (Cohen et al., 2007); network reliability optimization (Kroese et al., 2007a); vehicle routing optimization with stochastic demands (Chepuri and Homem-de-Mello, 2005); power system combinatorial optimization problems (Ernst et al., 2007); and neural and reinforcement learning (Lörincza et al., 2008; Menache et al., 2005; Unveren and Acan, 2007; Wu and Fyfe, 2008).

A tutorial on the CE method is given in de Boer et al. (2005). A comprehensive treatment can be found in Rubinstein and Kroese (2004); see also Rubinstein and Kroese (2007, Chapter 8). The CE method homepage is www.cemethod.org.

2. From estimation to optimization

The CE method can be applied to two types of problems:

1. *Estimation*: Estimate $\ell = \mathbb{E}[H(\mathbf{X})]$, where \mathbf{X} is a random object taking values in some set \mathcal{X} and H is a function on \mathcal{X} . An important special case is the estimation of a probability $\ell = \mathbb{P}(S(\mathbf{X}) \geq \gamma)$, where S is another function on \mathcal{X} .
2. *Optimization*: Optimize (that is, maximize or minimize) a given objective function $S(\mathbf{x})$ over all $\mathbf{x} \in \mathcal{X}$. S can be either a known or a *noisy* function. In the latter case the objective function needs to be estimated, e.g., via simulation.

In this section we review the CE method as an adaptive importance sampling method for rare-event probability estimation, and in the next section we show how the CE estimation algorithm leads naturally to an optimization heuristic.

2.1. Cross-entropy for rare-event probability estimation

Consider the estimation of the probability

$$\ell = \mathbb{P}(S(\mathbf{X}) \geq \gamma) = \mathbb{E} [I_{\{S(\mathbf{X}) \geq \gamma\}}] = \int I_{\{S(\mathbf{x}) \geq \gamma\}} f(\mathbf{x}; \mathbf{u}) d\mathbf{x}, \quad (1)$$

where S is a real-valued function, γ is a threshold or level parameter, and the random variable \mathbf{X} has probability density function (pdf) $f(\cdot; \mathbf{u})$, which is parameterized by a finite-dimensional real vector \mathbf{u} . We write $\mathbf{X} \sim f(\cdot; \mathbf{u})$. If \mathbf{X} is a discrete variable, simply replace the integral in (1) by a sum. We are interested in the case where ℓ is a *rare-event probability*; that is, a very small probability, say, less than 10^{-4} . Let g be

another pdf such that $g(\mathbf{x}) = 0 \Rightarrow H(\mathbf{x})f(\mathbf{x}; \mathbf{u}) = 0$ for every \mathbf{x} . Using the pdf g we can represent ℓ as

$$\ell = \int \frac{f(\mathbf{x}; \mathbf{u})I_{\{S(\mathbf{x}) \geq \gamma\}}}{g(\mathbf{x})} g(\mathbf{x}) d\mathbf{x} = \mathbb{E} \left[\frac{f(\mathbf{X}; \mathbf{u})I_{\{S(\mathbf{X}) \geq \gamma\}}}{g(\mathbf{X})} \right], \quad \mathbf{X} \sim g. \quad (2)$$

Consequently, if $\mathbf{X}_1, \dots, \mathbf{X}_N$ are independent random vectors with pdf g , written as $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} g$, then

$$\hat{\ell} = \frac{1}{N} \sum_{k=1}^N I_{\{S(\mathbf{X}_k) \geq \gamma\}} \frac{f(\mathbf{X}_k; \mathbf{u})}{g(\mathbf{X}_k)} \quad (3)$$

is an unbiased estimator of ℓ : a so-called *importance sampling estimator*. It is well known (see, for example, (Rubinstein and Kroese, 2007, p. 132)) that the optimal importance sampling pdf (that is, the pdf g^* for which the variance of $\hat{\ell}$ is minimal) is the density of \mathbf{X} conditional on the event $S(\mathbf{X}) \geq \gamma$; that is,

$$g^*(\mathbf{x}) = \frac{f(\mathbf{x}; \mathbf{u})I_{\{S(\mathbf{x}) \geq \gamma\}}}{\ell}. \quad (4)$$

The idea of the CE method is to choose the importance sampling pdf g from within the parametric class of pdfs $\{f(\cdot; \mathbf{v}), \mathbf{v} \in \mathcal{V}\}$ such that the Kullback–Leibler divergence between the optimal importance sampling pdf g^* and g is minimal. The Kullback–Leibler divergence between g^* and g is given by

$$\mathcal{D}(g^*, g) = \int g^*(\mathbf{x}) \ln \frac{g^*(\mathbf{x})}{g(\mathbf{x})} d\mathbf{x} = \mathbb{E} \left[\ln \frac{g^*(\mathbf{X})}{g(\mathbf{X})} \right], \quad \mathbf{X} \sim g^*. \quad (5)$$

The CE minimization procedure then reduces to finding an optimal reference parameter vector, \mathbf{v}^* say, by cross-entropy minimization:

$$\begin{aligned} \mathbf{v}^* &= \underset{\mathbf{v}}{\operatorname{argmin}} \mathcal{D}(g^*, f(\cdot; \mathbf{v})) \\ &= \underset{\mathbf{v}}{\operatorname{argmax}} \mathbb{E}_{\mathbf{u}} I_{\{S(\mathbf{X}) \geq \gamma\}} \ln f(\mathbf{X}; \mathbf{v}) \\ &= \underset{\mathbf{v}}{\operatorname{argmax}} \mathbb{E}_{\mathbf{w}} I_{\{S(\mathbf{X}) \geq \gamma\}} \ln f(\mathbf{X}; \mathbf{v}) \frac{f(\mathbf{X}; \mathbf{u})}{f(\mathbf{X}; \mathbf{w})}, \end{aligned} \quad (6)$$

where \mathbf{w} is any reference parameter and the subscript in the expectation operator indicates the density of \mathbf{X} . This \mathbf{v}^* can be estimated via the stochastic counterpart of (6):

$$\hat{\mathbf{v}} = \underset{\mathbf{v}}{\operatorname{argmax}} \frac{1}{N} \sum_{k=1}^N I_{\{S(\mathbf{X}_k) \geq \gamma\}} \frac{f(\mathbf{X}_k; \mathbf{u})}{f(\mathbf{X}_k; \mathbf{w})} \ln f(\mathbf{X}_k; \mathbf{v}), \quad (7)$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \mathbf{w})$. The optimal parameter $\hat{\mathbf{v}}$ in (7) can often be obtained in explicit form, in particular when the class of sampling distributions forms an *exponential family* (Rubinstein and Kroese, 2007, pp. 319–320). Indeed, analytical updating formulas can be found whenever explicit expressions for the *maximal likelihood estimators* of the parameters can be found (de Boer et al., 2005, pp. 36).

A complication in solving (7) is that for a rare-event probability ℓ most or all of the indicators $I_{\{S(\mathbf{X}) \geq \gamma\}}$ in (7) are zero, and the maximization problem becomes useless. In that case a *multi-level* CE procedure is used, where a sequence of reference parameters $\{\hat{\mathbf{v}}_t\}$ and levels $\{\hat{\gamma}_t\}$ is constructed with the goal that the former converges to \mathbf{v}^* and the latter to γ . At each iteration t we simulate N independent random variables $\mathbf{X}_1, \dots, \mathbf{X}_N$ from the currently estimated importance sampling density $f(\cdot; \hat{\mathbf{v}}_{t-1})$ and let $\hat{\gamma}_t$ be the $(1 - \varrho)$ -quantile of the performance values $S(\mathbf{X}_1), \dots, S(\mathbf{X}_N)$, where ϱ is a user specified parameter called the *rarity parameter*. We then update the value of $\hat{\mathbf{v}}_{t-1}$ to $\hat{\mathbf{v}}_t$, where $\hat{\mathbf{v}}_t$ is calculated using likelihood maximization (or equivalently cross-entropy minimization) based on the $N^e = \lceil \varrho N \rceil$ random variables for which $S(\mathbf{X}_i) \geq \hat{\gamma}_t$.

This leads to the following algorithm (Rubinstein and Kroese, 2007, p. 238).

Algorithm 2.1 (CE Algorithm for Rare-Event Estimation) Given the sample size N and the parameter ϱ , execute the following steps.

1. Define $\hat{\mathbf{v}}_0 = \mathbf{u}$. Let $N^e = \lceil \varrho N \rceil$. Set $t = 1$ (iteration counter).
2. Generate $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}}_{t-1})$. Calculate $S_i = S(\mathbf{X}_i)$ for all i , and order these from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t$ be the sample $(1 - \varrho)$ -quantile of performances; that is, $\hat{\gamma}_t = S_{(N-N^e+1)}$. If $\hat{\gamma}_t > \gamma$, reset $\hat{\gamma}_t$ to γ .
3. Use the **same** sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ to solve the stochastic program (7), with $\mathbf{w} = \hat{\mathbf{v}}_{t-1}$. Denote the solution by $\hat{\mathbf{v}}_t$.
4. If $\hat{\gamma}_t < \gamma$, set $t = t + 1$ and reiterate from Step 2; otherwise, proceed with Step 5.
5. Let $T = t$ be the final iteration counter. Generate $\mathbf{X}_1, \dots, \mathbf{X}_{N_1} \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}}_T)$ and estimate ℓ via importance sampling, as in (3) with $\mathbf{u} = \hat{\mathbf{v}}_T$.

We now show how this estimation algorithm leads naturally to a simple optimization heuristic.

2.2. Cross-entropy method for optimization

To see how Algorithm 2.1 can be used for optimization purposes, suppose that the goal is to find the maximum of $S(\mathbf{x})$ over a given set \mathcal{X} . Assume, for simplicity, that there is only one maximizer \mathbf{x}^* . Denote the maximum by γ^* , so that

$$S(\mathbf{x}^*) = \gamma^* = \max_{\mathbf{x} \in \mathcal{X}} S(\mathbf{x}). \quad (8)$$

We can now associate with the above optimization problem the estimation of the probability $\ell = \mathbb{P}(S(\mathbf{X}) \geq \gamma)$, where \mathbf{X} has some probability density $f(\mathbf{x}; \mathbf{u})$ on \mathcal{X} (for example corresponding to the uniform distribution on \mathcal{X}) and γ is close to the unknown γ^* . Typically, ℓ is a rare-event probability, and the multi-level CE approach of Algorithm 2.1 can be used to find an important sampling distribution that concentrates all its mass in a neighborhood of the point \mathbf{x}^* . Sampling from such a distribution thus produces optimal or near-optimal states. Note that, in contrast to the rare-event simulation setting, the final level $\gamma = \gamma^*$ is generally not known in advance, but the CE method for optimization produces a sequence of levels $\{\hat{\gamma}_t\}$ and reference parameters $\{\hat{\mathbf{v}}_t\}$ such that ideally the former tends to the optimal γ^* and the latter to the optimal reference vector \mathbf{v}^* corresponding to the point mass at \mathbf{x}^* ; see, e.g., (Rubinstein and Kroese, 2007, p. 251).

Algorithm 2.2 (CE Algorithm for Optimization)

1. Choose an initial parameter vector $\hat{\mathbf{v}}_0$. Let $N^e = \lceil \varrho N \rceil$. Set $t = 1$ (level counter).
2. Generate $\mathbf{X}_1, \dots, \mathbf{X}_N \sim_{\text{iid}} f(\cdot; \hat{\mathbf{v}}_{t-1})$. Calculate the performances $S(\mathbf{X}_i)$ for all i , and order them from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t$ be the sample $(1 - \varrho)$ -quantile of performances; that is, $\hat{\gamma}_t = S_{(N-N^e+1)}$.
3. Use the **same** sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ and solve the stochastic program

$$\max_{\mathbf{v}} \frac{1}{N} \sum_{k=1}^N I_{\{S(\mathbf{X}_k) \geq \hat{\gamma}_t\}} \ln f(\mathbf{X}_k; \mathbf{v}). \quad (9)$$

Denote the solution by $\hat{\mathbf{v}}_t$.

4. If some stopping criterion is met, stop; otherwise, set $t = t + 1$, and return to Step 2.

Note that the estimation Step 5 of Algorithm 2.1 is missing in Algorithm 2.2, because in the optimization setting we are not interested in estimating ℓ per se. For the same reason the likelihood ratio term $f(\mathbf{X}_k; \mathbf{u})/f(\mathbf{X}_k; \hat{\mathbf{v}}_{t-1})$ in (7) is missing in (9).

To run the algorithm one needs to propose a class of parametric sampling densities $\{f(\cdot; \mathbf{v}), \mathbf{v} \in \mathcal{V}\}$, the initial vector $\hat{\mathbf{v}}_0$, the sample size N , the rarity parameter ϱ , and a stopping criterion. Of these the most challenging is the selection of an appropriate class of parametric sampling densities $\{f(\cdot; \mathbf{v}), \mathbf{v} \in \mathcal{V}\}$. Typically, there is not a unique parametric family and the selection is guided by the following competing objectives. First, the class $\{f(\cdot; \mathbf{v}), \mathbf{v} \in \mathcal{V}\}$ has to be flexible enough to include a reasonable parametric approximation to the optimal importance sampling density (4) for the estimation of the associated rare-event probability ℓ . Second, each density $f(\cdot; \mathbf{v})$ has to be simple enough to allow fast random variable generation and closed-form solutions to the (weighted) maximum likelihood estimation program (9). In many cases these two competing objectives are reconciled by using a standard statistical model for $f(\cdot; \mathbf{v})$, such as the multivariate Bernoulli or Gaussian densities with independent components of the vector $\mathbf{X} \sim f(\cdot; \mathbf{v})$. However, in some cases it is beneficial to consider the more complicated Bernoulli or Gaussian mixture models, in which the estimation program (9) is carried out using the EM algorithm (see Section 3.2). In the examples that follow we primarily use the multivariate Bernoulli and Gaussian densities for combinatorial and continuous optimization problems, respectively. Note, however, that special parameterizations may be needed for problems such as the traveling salesman problem, where the optimization is carried out over a set of possible permutations; see Rubinstein and Kroese (2004) for more details.

In applying Algorithm 2.2, the following *smoothed updating* has proven useful. Let $\tilde{\mathbf{v}}_t$ be the solution to (9) and $0 \leq \alpha \leq 1$ be a *smoothing parameter*, then for the next iteration we take the parameter vector

$$\hat{\mathbf{v}}_t = \alpha \tilde{\mathbf{v}}_t + (1 - \alpha) \hat{\mathbf{v}}_{t-1}. \quad (10)$$

Other modifications can be found in Kroese et al. (2006) and Rubinstein and Kroese (2004, 2007). The effect that smoothing has on convergence is discussed in detail in Costa et al. (2007). In particular, it is shown that for discrete binary optimization

problems with nonconstant smoothing $\hat{v}_t = \alpha_t \tilde{v}_t + (1 - \alpha_t) \hat{v}_{t-1}$, $\alpha_t \in [0, 1]$ the optimal solution is generated with probability 1 if the smoothing sequence $\{\alpha_t\}$ satisfies:

$$\sum_{t=1}^{\infty} \prod_{k=1}^t (1 - \alpha_k)^n = \infty,$$

where n is the length of the binary vector \mathbf{x} . Other convergence results can be found in [Rubinstein and Kroese \(2004\)](#) and [Margolin \(2005\)](#). Finally, significant speedup can be achieved by using a *parallel* implementation of CE ([Evans et al., 2007](#)).

3. Applications to combinatorial optimization

When the state space \mathcal{X} is finite, the optimization problem (8) is often referred to as a *discrete* or *combinatorial optimization* problem. For example, \mathcal{X} could be the space of combinatorial objects such as binary vectors, trees, paths through graphs, permutations, etc. To apply the CE method, one needs to first specify a convenient parameterized random mechanism to generate objects \mathbf{X} in \mathcal{X} . An important example is where $\mathbf{X} = (X_1, \dots, X_n)$ has independent components such that $X_i = j$ with probability $v_{i,j}$, $i = 1, \dots, n$, $j = 1, \dots, m$. In that case, the solution of the program in (9) at the t th iteration is

$$\hat{v}_{t,i,j} = \frac{\sum_{k=1}^N \mathbf{I}_{\{S(\mathbf{X}_k) \geq \hat{y}_t\}} \mathbf{I}_{\{X_{k,i} = j\}}}{\sum_{k=1}^N \mathbf{I}_{\{S(\mathbf{X}_k) \geq \hat{y}_t\}}}, \quad i = 1, \dots, n, \quad j = 1, \dots, m, \quad (11)$$

where $\mathbf{X}_1, \dots, \mathbf{X}_N$ are independent copies of $\mathbf{X} \sim \{\hat{v}_{t-1,i,j}\}$ and $X_{k,i}$ is the i th element of \mathbf{X}_k (see [\(de Boer et al., 2005, p. 56\)](#)). We call the set of vectors satisfying $S(\mathbf{X}_k) \geq \hat{y}_t$ the *elite sample* at iteration t . Thus, the updated probability $\hat{v}_{t,i,j}$ is simply the number of elite samples for which the i th component is equal to j , divided by the total number of elite samples. Note that (11) is a direct consequence, via (9), of the cross-entropy minimization program (6).

A possible stopping rule for combinatorial optimization problems is to stop when the overall best objective value does not change over a number of iterations. Alternatively, one could stop when the sampling distribution has “degenerated” enough. For example, when in (11) the $\{\hat{v}_{t,i,j}\}$ differ less than some small $\epsilon > 0$ from the $\{\hat{v}_{t-1,i,j}\}$.

3.1. Knapsack packing problem

A well-known NP-complete combinatorial optimization problem is the 0–1 knapsack problem ([Kellerer et al., 2004](#)) defined as:

$$\begin{aligned} & \max_{\mathbf{x}} \sum_{j=1}^n p_j x_j, \quad x_j \in \{0, 1\}, \quad j = 1, \dots, n, \\ & \text{subject to : } \sum_{j=1}^n w_{i,j} x_j \leq c_i, \quad i = 1, \dots, m. \end{aligned} \quad (12)$$

Here $\{p_j\}$ and $\{w_{i,j}\}$ are positive weights, $\{c_i\}$ are positive cost parameters, and $\mathbf{x} = (x_1, \dots, x_n)$. In order to define a single objective function, we adopt the penalty function approach and define

$$S(\mathbf{x}) \stackrel{\text{def}}{=} \beta \sum_{i=1}^m I_{\left\{ \sum_j w_{i,j} x_j > c_i \right\}} + \sum_{j=1}^n p_j x_j,$$

where $\beta = -\sum_{j=1}^n p_j$. In this way, $S(\mathbf{x}) \leq 0$ if one of the inequality constraints is not satisfied and $S(\mathbf{x}) = \sum_{j=1}^n p_j x_j$ if all of the constraints are satisfied. As a particular example, consider the *Sento1.dat* knapsack problem given in the appendix of (Senju and Toyoda, 1968). The problem has 30 constraints and 60 variables. Since the solution vector \mathbf{x} is binary, a simple choice for the sampling density in Step 2 of Algorithm 2.2 is the multivariate Bernoulli density $f(\mathbf{x}; \mathbf{v}) = \prod_{j=1}^n v_j^{x_j} (1 - v_j)^{1-x_j}$. We apply Algorithm 2.2 to this particular problem with $N = 10^3$ and $N^e = 20$, $\hat{\mathbf{v}}_0 = (1/2, \dots, 1/2)$. Note that we do not use any smoothing for $\hat{\mathbf{v}}_t$ (that is, $\alpha = 1$ in (10)) and the solution $\hat{\mathbf{v}}_t$ of (9) at each iteration is given by

$$\hat{v}_{t,j} = \frac{\sum_{k=1}^N I_{\{\hat{S}(\mathbf{X}_k) \geq \hat{v}_t\}} X_{k,j}}{\sum_{k=1}^N I_{\{\hat{S}(\mathbf{X}_k) \geq \hat{v}_t\}}}, \quad j = 1, \dots, n, \quad (13)$$

where $X_{k,j}$ is the j th component of the k th random binary vector \mathbf{X} . In Step 4 of Algorithm 2.2 we stop the algorithm if $d_t = \max_{1 \leq j \leq n} \{\min\{\hat{v}_{t,j}, 1 - \hat{v}_{t,j}\}\} \leq 0.01$. Table 1 shows the typical evolution of the CE combinatorial optimization algorithm on the *Sento1.dat* problem. For each iteration t we recorded the threshold \hat{v}_t ,

Table 1

Typical evolution of the CE method on the knapsack problem. The last column shows the stopping value $d_t = \max_{1 \leq j \leq n} \{\min\{\hat{v}_{t,j}, 1 - \hat{v}_{t,j}\}\}$ for each t .

t	\hat{v}_t	Best score	d_t
1	-19,2450	-45,420	0.495
2	-11,9180	1056	0.455
3	-48,958	1877	0.395
4	-8953	3331	0.34
5	1186	3479	0.415
6	2039.5	4093	0.455
7	2836.5	4902	0.435
8	3791	5634	0.485
9	4410	5920	0.46
10	5063.5	6246	0.46
11	5561	6639	0.452
12	5994.5	6704	0.47
13	6465.5	6704	0.46
14	6626	6704	0.355
15	6677	6704	0.346
16	6704	6704	0

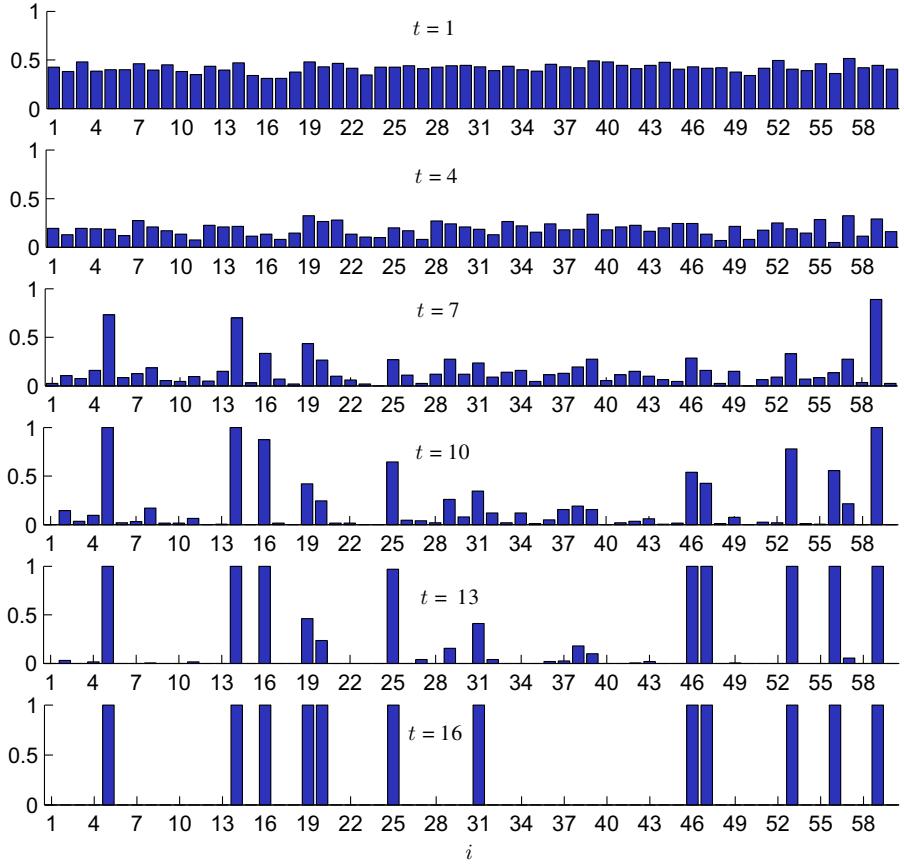


Fig. 1. The evolution of the probability vector \hat{v}_t .

the largest value of $S(\mathbf{X}_k)$ in the current population, and the value of the stopping criterion d_t .

The global maximum value for this problem is 6704. Figure 1 shows the evolution of the probability vector \hat{v}_t , which characterizes the multivariate Bernoulli distribution $f(\cdot; \hat{v}_t)$. Note that \hat{v}_t converges to a binary vector corresponding to the optimal solution.

3.2. Boolean satisfiability problem

Let $\mathbf{x} = (x_1, \dots, x_n)$ be a binary vector and suppose $c_1(\mathbf{x}), \dots, c_m(\mathbf{x})$ are m functions, called *clause functions*, such that $c_i : \{0,1\}^n \rightarrow \{0,1\}$. For given clauses, the objective is to find

$$\mathbf{x}^* = \operatorname{argmax}_{\mathbf{x} \in \{0,1\}^n} \prod_{i=1}^m c_i(\mathbf{x}). \quad (14)$$

That is, to find a vector \mathbf{x}^* satisfying all clauses simultaneously. This Boolean satisfiability (SAT) problem (Gu et al., 1997) is of importance in computer science and operations research, because any NP-complete problem, such as the traveling salesman problem, can be translated into a SAT problem in polynomial time.

We now show how one can apply Algorithm 2.2 to the SAT problem. First, note that the function $\prod_{i=1}^m c_i(\mathbf{x})$ in (14) takes on only the values 0 or 1, and is thus not suitable for the level approach in the CE method. A more suitable and equivalent formulation is:

$$\mathbf{x}^* = \operatorname{argmax}_{\mathbf{x} \in \{0,1\}^n} S(\mathbf{x}) \stackrel{\text{def}}{=} \operatorname{argmax}_{\mathbf{x} \in \{0,1\}^n} \sum_{i=1}^m c_i(\mathbf{x}), \quad (15)$$

where the function $S : \{0,1\}^n \rightarrow \{1, \dots, m\}$. In this way, the function S can be used to measure the distance from our goal by indicating the number of satisfied clauses (as opposed to simply indicating whether all clauses are satisfied or not). As a specific example we consider the SAT instance `uf75-01` from Hoos and Stützle (2000).

To illustrate the role of the sampling distribution, we first use the same method of generation as in the knapsack problem; that is, at each iteration t the j th component of \mathbf{X} is generated according to a $\text{Ber}(\hat{v}_{t,j})$ distribution, independently of all other components. The updating formula is thus (13). We run the CE algorithm with a sample size of $N = 10^4$ and a rarity parameter of $\varrho = 0.1$, giving $N^e = 10^3$ elite samples per iteration. We take $\hat{\mathbf{v}}_0 = (0.5, \dots, 0.5)$ and do not use smoothing ($\alpha = 1$). The upper panel of Fig. 2 shows the evolution of the method over the first 30 iterations. The figure shows the scores of the best and worst performing elite samples at each iteration t , together with the value $S(\mathbf{x}^*) = 325$. The optimization gets close to the maximal level 325 (indicated by a horizontal line), but stops at 323. The reason why the CE algorithm does not reach the maximum is that the sampling distribution is too simple.

Next, instead of the basic multivariate Bernoulli model for $f(\cdot; \mathbf{v})$, we consider the Bernoulli mixture model (see Section 4.2 for a discussion of parametric mixture models)

$$f(\mathbf{x}; \mathbf{v}) = \sum_{k=1}^K w_k \prod_{j=1}^n p_{k,j}^{x_j} (1 - p_{k,j})^{1-x_j},$$

where K is the number of mixture components, $\mathbf{w} = (w_1, \dots, w_K)$ are the weights ($w_k \geq 0$, $\sum_k w_k = 1$) associated with each component, each $\mathbf{p}_k = (p_{k,1}, \dots, p_{k,n})$ is a vector of probabilities, and $\mathbf{v} = (\mathbf{w}, \mathbf{p}_1, \dots, \mathbf{p}_K)$ collects all the unknown parameters (we assume that K is given). The greater flexibility in the parametric model comes at a cost—the maximum likelihood program (9) in Step 3 of Algorithm 2.2 no longer has a simple closed-form solution. Nevertheless, an approximate solution of (9) can be obtained using the EM algorithm of (Dempster et al., 1977), where the initial starting point for the EM routine is chosen at random from the elite vectors. The lower panel of Fig. 2 shows the evolution of this CE algorithm using $K = 6$ components for the Bernoulli mixture density and the same algorithmic parameters ($N = 10^4, \varrho = 0.1, \alpha = 1, \hat{\mathbf{v}}_0 = (0.5, \dots, 0.5)$). With this setup the algorithm quickly

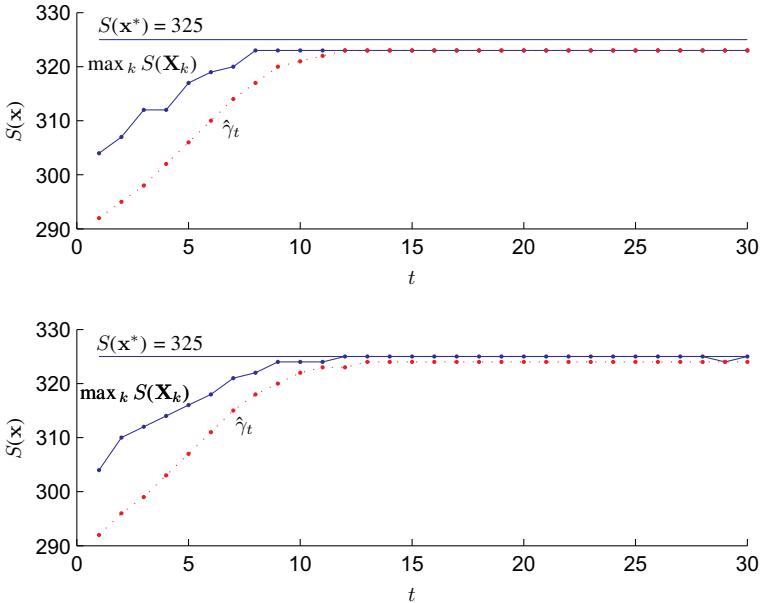


Fig. 2. Evolution of the CE algorithm on the uF75-01 SAT instance. The upper panel corresponds to the case where $f(\mathbf{x}; \mathbf{v})$ is a multivariate Bernoulli density and the lower panel corresponds to the case where $f(\mathbf{x}; \mathbf{v})$ is Bernoulli mixture model with $K = 6$ components. Both panels show the best score, $\max_k S(\bar{X}_k)$, and $\hat{\gamma}_t$ for each iteration t .

reaches the desired level of 325, justifying the extra computational cost of fitting a parametric mixture model via the EM algorithm.

3.3. Network planning problem

An important integer optimization problem is the *Network Planning Problem* (NPP). Here a network is represented as an undirected graph with edges (links) that may fail with a given probability. The objective of the NPP is to optimally purchase a collection of edges, subject to a fixed budget, so as to maximize the *network reliability*—defined as the probability that all nodes in a given set of nodes are connected by functioning edges. Each edge comes with a pre-specified cost and reliability (probability that it works). The NPP has applications in engineering (Wang et al., 2009), telecommunications, transportation, energy supply systems (Leite de Silva et al., 2010; Kothari and Kroese, 2009), computer, and social networking (Hintsanen et al., 2010). The difficulty in solving this problem derives from the following aspects of the optimization:

1. Typically the computation of the reliability of large networks is a difficult $\#P$ -complete computational problem, which requires either Monte Carlo simulation or sharp bounds on the reliability (Rubino, 1998; Won and Karray, 2010). In the Monte Carlo case the NPP becomes a noisy optimization problem, that is, the objective function values are estimated from simulation.

2. For highly reliable networks, the network reliability is a rare-event probability. In such cases Crude Monte Carlo simulation is impractical and one has to resort to sophisticated rare-event probability estimation methods (Rubino and Tuffin, 2009).
3. Even if the reliability could somehow be computed easily, thus dispensing with the problems above, the NPP remains an NP-hard 0–1 knapsack optimization problem with a nonlinear objective function. In response to this, various optimization heuristics have been developed, such as simulated annealing (Cancela and Urquhart, 1995) and genetic algorithms (Dengiz et al., 1997; Reichelt et al., 2007).
4. Finally, the problem is constrained and for most optimization heuristics like the simulated annealing and genetic algorithms, the penalty function approach is inefficient.

Here we apply the CE method to tackle all of the above problems simultaneously. We now explain the implementation and mathematical details, largely following (Kroese et al., 2007a).

Let $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{K})$ be an undirected graph, where $\mathcal{V} = \{1, \dots, \dot{v}\}$ is the set of \dot{v} nodes (vertices), $\mathcal{E} = \{1, \dots, n\}$ is the set of n edges, and $\mathcal{K} \subseteq \mathcal{V}$ with $|\mathcal{K}| \geq 2$ is the set of *terminal nodes*. Sometimes we may refer to an edge by specifying the pair of nodes it connects, and write

$$\mathcal{E} = \{(\dot{v}_i, \dot{v}_j), \dots, (\dot{v}_k, \dot{v}_l)\}, \quad \dot{v}_i, \dot{v}_j, \dot{v}_k, \dot{v}_l \in \mathcal{V}.$$

For example, in Fig. 3 the edge set is given by

$$\mathcal{E} = \{(1,2), (1,3), (1,6), (2,3), (2,4), (3,4), (3,6), (4,5), (5,6)\},$$

and the set of terminal nodes is $\{2, 5\}$. For each $e = 1, \dots, n$ let $B_e \sim \text{Ber}(p_e)$ be a Bernoulli random variable with success probability p_e indicating whether the e th link is operational. In other words, the event $\{B_e = 1\}$ corresponds to edge e being operational and the event $\{B_e = 0\}$ corresponds to the edge being nonoperational.

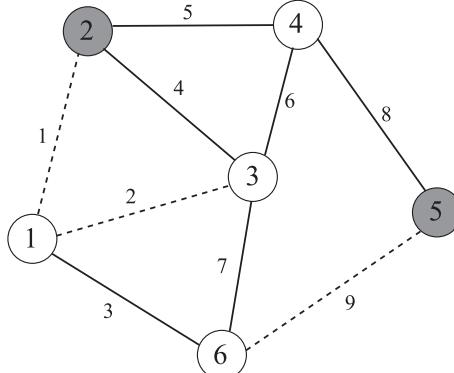


Fig. 3. A reliability network with six nodes and nine edges or links. The network works if the two terminal nodes (filled circles) are connected by functioning links. Failed links are indicated by dashed lines.

It follows that each of the 2^n states of the network can be represented by a binary vector $\mathbf{B} = (B_1, \dots, B_n)$. For example, the state of the network on Fig. 3 can be represented as $\mathbf{B} = (0,0,1,1,1,1,1,0)$. Let $\mathcal{E}[\mathbf{B}]$ denote the set of operational edges corresponding to \mathbf{B} . For example, the set of operational edges on Fig. 3 is

$$\mathcal{E}[\mathbf{B}] = \{(1,6), (2,3), (2,4), (3,4), (3,6), (4,5)\} \equiv \{3,4,5,6,7,8\}.$$

The size of the set of edges $\mathcal{E}[\mathbf{B}]$ is $|\mathcal{E}[\mathbf{B}]| = B_1 + \dots + B_n$. The *reliability* of the network is the probability that all of the nodes in \mathcal{K} are connected by the set $\mathcal{E}[\mathbf{B}]$ of operational edges. For a given set of edges \mathcal{E} we define the *structure function* Φ as

$$\Phi[\mathcal{E}] = \begin{cases} 1, & \text{if } \mathcal{E} \text{ contains a path connecting the nodes in } \mathcal{K}, \\ 0, & \text{if } \mathcal{E} \text{ does not contain a path connecting the nodes in } \mathcal{K}. \end{cases}$$

We can write the reliability of the network as

$$r = \sum_{\mathbf{b}} \Phi(\mathcal{E}[\mathbf{b}]) \prod_{e=1}^n p_e^{b_e} (1 - p_e)^{1-b_e} = \mathbb{E}\Phi(\mathcal{E}[\mathbf{B}]),$$

where $B_e \sim \text{Ber}(p_e)$, independently for all $e = 1, \dots, n$. Denote by $\mathbf{x} = (x_1, \dots, x_n)$ the *purchase vector* with $(e = 1, \dots, n)$

$$x_e = \begin{cases} 1, & \text{if link } e \text{ is purchased,} \\ 0, & \text{if link } e \text{ is not purchased.} \end{cases}$$

Let us denote by $\mathbf{c} = (c_1, \dots, c_n)$ the vector of link costs. Define

$$\mathbf{B}(\mathbf{x}) = (B_1 x_1, \dots, B_n x_n),$$

$$S(\mathbf{x}) = \mathbb{E}\Phi(\mathcal{E}[\mathbf{B}(\mathbf{x})]),$$

where $S(\mathbf{x})$ is the reliability for a given purchase vector \mathbf{x} . Then, the NPP can be written as:

$$\begin{aligned} & \max_{\mathbf{x}} S(\mathbf{x}) \\ & \text{subject to : } \sum_e x_e c_e \leq C_{\max}, \end{aligned} \tag{16}$$

where C_{\max} is the available budget. We next address the problem of estimating $S(\mathbf{x})$ for highly reliable networks using Monte Carlo simulation.

3.3.1. Permutation Monte Carlo and merge process

In principle, any of the sophisticated rare-event probability estimation methods presented in Rubino and Tuffin (2009) can be used to estimate $S(\mathbf{x})$. Here we employ the merge process (MP) of (Elperin et al., 1991). Briefly, (Elperin et al., 1991) view the static network as a snapshot of a dynamical network, in which the edges evolve over time—starting failed and eventually becoming operational (or, alternatively, starting operational and eventually failing). They exploit the fact that

one can compute exactly the reliability of the network, given the order in which the links of the network become operational, and suggest a conditional Monte Carlo method as a variance reduction tool: generate random permutations (indicating the order in which the links become operational) and then evaluate the unreliability for each permutation.

Suppose $\mathcal{E}_x = \mathcal{E}[x] = \{e_1, \dots, e_m\} \subseteq \mathcal{E}$ is the set of purchased links. The corresponding link reliabilities are p_{e_1}, \dots, p_{e_m} . Define $\lambda_e = -\log(1 - p_e)$.

Algorithm 3.1 (Merge Process for Graph $\mathcal{G}(\mathcal{V}, \mathcal{E}_x, \mathcal{K})$)

1. Generate $E_i \sim \text{Exp}(\lambda_{e_i})$ independently for all purchased links $e_i \in \mathcal{E}_x$. Sort the sequence E_1, \dots, E_m to obtain the permutation $(\pi(1), \dots, \pi(m))$ such that

$$E_{\pi(1)} < E_{\pi(2)} < \dots < E_{\pi(m)}.$$

Let \mathcal{D} be a dynamic ordered list of nonoperational links. Initially, set

$$\mathcal{D} = (e_{\pi(1)}, e_{\pi(2)}, \dots, e_{\pi(m)}).$$

Compute

$$\lambda_1^* = \sum_{e \in \mathcal{D}} \lambda_e$$

and set the counter $b = 1$.

2. Let e^* be the first link in the list \mathcal{D} . Remove link e^* from \mathcal{D} and add it to the set \mathcal{U} (which need not be ordered). Let $\mathcal{G}_b \equiv \mathcal{G}(\mathcal{V}, \mathcal{U}, \mathcal{K})$ be the network in which all edges in \mathcal{U} are working and the rest of the edges (all edges in \mathcal{D}) are failed.
3. If the network \mathcal{G}_b is operational go to Step 6; otherwise, continue to Step 4.
4. Identify the *connected component* of which link e^* is a part. Find all links in \mathcal{D} in this connected component. These are called *redundant links*, because they connect nodes that are already connected. Denote the set of redundant links by \mathcal{R} .
5. Remove the links in \mathcal{R} from \mathcal{D} . Compute

$$\lambda_{b+1}^* = \lambda_b^* - \lambda_{e^*} - \sum_{e \in \mathcal{R}} \lambda_e = \sum_{e \in \mathcal{D}} \lambda_e.$$

Increment $b = b + 1$ and repeat from Step 2.

6. Let A be the $b \times b$ square matrix:

$$A = \begin{pmatrix} -\lambda_1^* & \lambda_1^* & 0 & \dots & 0 \\ 0 & -\lambda_2^* & \lambda_2^* & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -\lambda_{b-1}^* & \lambda_{b-1}^* \\ 0 & \dots & 0 & 0 & -\lambda_b^* \end{pmatrix},$$

and let $A^* = (A_{i,j}^*) = e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!}$ be the matrix exponential of A . Output the unbiased estimator of the unreliability of the network

$$Z = \sum_{j=1}^b A_{1,j}^*$$

For the NPP we use the following unbiased estimator of the reliability $S(\mathbf{x})$:

$$\widehat{S}(\mathbf{x}) = 1 - \frac{1}{M} \sum_{k=1}^M Z_k, \quad (17)$$

where Z_1, \dots, Z_M are independent realizations of Z from Algorithm 3.1. Note that if $\mathcal{E}_{\mathbf{x}}$ does not contain enough operational links to connect the nodes in \mathcal{K} , then the network reliability is trivially equal to 0.

3.3.2. Sampling with a budget constraint

While it is possible to incorporate the budget constraint via a penalty function added to (17), here we take a more direct approach, in which we generate a sample of n dependent Bernoulli random variables such that the constraint $\sum_e c_e X_e \leq C_{\max}$ is satisfied. In other words, in Step 2 of Algorithm 2.2 we sample random purchase vectors $\mathbf{X}_1, \dots, \mathbf{X}_N$ from a parametric density $f(\cdot; \mathbf{v}), \mathbf{v} = (v_1, \dots, v_n)$, that takes into account the budget constraint in (16) and is implicitly defined via the following algorithm (Kroese et al., 2007a).

Algorithm 3.2 (Purchase vector generation) Given the vector \mathbf{v} , set $i = 1$ and execute the following steps.

1. Generate $U_1, \dots, U_n \stackrel{\text{iid}}{\sim} \text{U}(0,1)$ and let π be the permutation which satisfies $U_{\pi(1)} < U_{\pi(2)} < \dots < U_{\pi(n)}$.
2. If $c_{\pi(i)} + \sum_{e=1}^{i-1} c_{\pi(e)} X_{\pi(e)} \leq C_{\max}$, generate $U \sim \text{U}(0,1)$ and set $X_{\pi(i)} = \mathbf{I}_{\{U < v_{\pi(i)}\}}$; otherwise, set $X_{\pi(i)} = 0$.
3. If $i < n$, increment $i = i + 1$ and repeat from Step 2; otherwise, deliver the random purchase vector $\mathbf{X} = (X_1, \dots, X_n)$.

We can interpret \mathbf{v} as a vector of purchase probabilities. The aim of the CE optimization Algorithm 2.2 is then to construct a sequence of probabilities $\{\hat{\mathbf{v}}_t, t = 0, 1, 2, 3\}$ that converges to a degenerate (binary) vector \mathbf{x}^* corresponding to the optimal purchase vector. The updating of $\hat{\mathbf{v}}_t$ at each iteration is given by (9) with S replaced by the estimated \widehat{S} . The solution of (9) is:

$$\tilde{v}_{t,j} = \frac{\sum_{k=1}^N \mathbf{I}_{\{\widehat{S}(\mathbf{X}_k) \geq \hat{y}_t\}} X_{k,j}}{\sum_{k=1}^N \mathbf{I}_{\{\widehat{S}(\mathbf{X}_k) \geq \hat{y}_t\}}}, \quad j = 1, \dots, n. \quad (18)$$

For clarity we now restate the CE optimization algorithm as applied to the NPP.

Algorithm 3.3 (CE Optimization for NPP)

1. Let $\hat{\mathbf{v}}_0 = (1/2, \dots, 1/2)$. Let $N^e = \lceil \varrho N \rceil$, where ϱ is the user-specified rarity parameter. Set $t = 1$ (level counter).

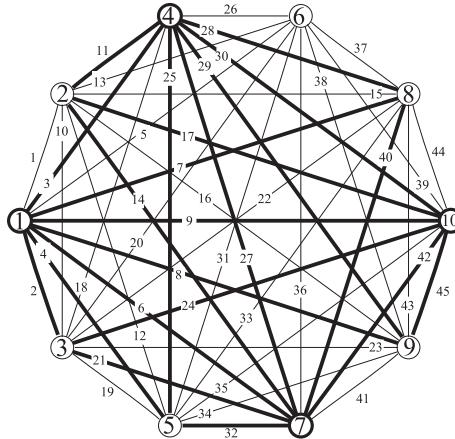


Fig. 4. A complete graph with ten nodes and 45 edges.

2. Generate independent purchase vectors $\mathbf{X}_1, \dots, \mathbf{X}_N$ using Algorithm 3.2 with $\mathbf{v} = \hat{\mathbf{v}}_t$. For each purchase vector estimate the reliability $\hat{S}(\mathbf{X}_t)$ using the merge process estimator (17) and rank the reliabilities from smallest to largest: $S_{(1)} \leq \dots \leq S_{(N)}$. Let $\hat{\gamma}_t$ be the sample $(1 - \varrho)$ -quantile of reliabilities; that is, $\hat{\gamma}_t = S_{(N-N^e+1)}$.
3. Use the same sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ to estimate the purchase probabilities via (18).
4. Smooth the purchase probabilities in Step 3 to obtain $\hat{\mathbf{v}}_t$ with

$$\hat{v}_{t,j} = \alpha \tilde{v}_{t,j} + (1 - \alpha) \hat{v}_{t-1,j}, \quad j = 1, \dots, n.$$

5. If

$$\max_{1 \leq j \leq n} \{\min\{\hat{v}_{t,j}, 1 - \hat{v}_{t,j}\}\} \leq \epsilon$$

for some small ϵ , say $\epsilon = 10^{-2}$, stop; otherwise, set $t = t + 1$ and repeat from Step 2.

We now illustrate the method on the complete graph K_{10} given in Fig. 4.

The costs and reliabilities for the graph were generated in the (quite arbitrary) pseudo-random fashion as follows.

Algorithm 3.4 (Generation of Costs and Reliabilities for a K_{10} graph)

```

Set  $i = 0$  and  $j = 1$ 
while  $j \leq 45$  do
   $i \leftarrow i + 1$ 
   $b = 1 - (7654321 \bmod (987 + i)) / 10^5$ 
  if  $b > 0.99$  and  $b \neq p_k$  for all  $k$  then
     $p_j = b$ 
     $c_j = 20 / \exp(8\sqrt{1 - b})$ 
     $j \leftarrow j + 1$ 
  end if
end while

```

We let $\mathcal{K} = \{1, 4, 7, 10\}$ and $C_{\max} = 250$. We apply Algorithm 3.3 with $N = 100$, $N^e = 10$ ($\varrho = 0.1$), $\alpha = 1/2$, and $\epsilon = 10^{-2}$. For the merge process estimator (17) we used $M = 100$. The estimated overall purchase vector corresponds to the following edges:

$$\mathcal{E}[\mathbf{x}^*] = \{2, 3, 4, 6, 7, 8, 9, 11, 14, 17, 21, 24, 25, 27, 28, 29, 30, 32, 40, 42, 45\},$$

with a total cost of 241.2007. These edges are drawn thicker on Fig. 4. Using $M = 10^5$ samples in (17) we obtained the estimate 4.80×10^{-16} (with an estimated relative error of 1.7%) of the corresponding unreliability of the purchased network.

4. Continuous optimization

When the state space is continuous, in particular when $\mathcal{X} = \mathbb{R}^n$, the optimization problem is often referred to as a *continuous optimization* problem. The CE sampling distribution on \mathbb{R}^n can be quite arbitrary and does not need to be related to the function that is being optimized. The generation of a random vector $\mathbf{X} = (X_1, \dots, X_n) \in \mathbb{R}^n$ in Step 2 of Algorithm 2.2 is most easily performed by drawing the n coordinates independently from some two-parameter distribution. In most applications a normal (Gaussian) distribution is employed for each component. Thus, the sampling density $f(\cdot; \mathbf{v})$ of \mathbf{X} is characterized by a vector of means $\boldsymbol{\mu}$ and a vector of variances $\boldsymbol{\sigma}^2$ (and we may write $\mathbf{v} = (\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$). The choice of the normal distribution is motivated by the availability of fast normal random number generators on modern statistical software and the fact that the maximum likelihood maximization (or cross-entropy minimization) in (9) yields a very simple solution—at each iteration of the CE algorithm the parameter vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are the vectors of sample means and sample variance of the elements of the set of N^e best performing vectors (that is, the elite set); see, for example, Kroese et al. (2006). In summary, the CE method for continuous optimization with a Gaussian sampling density is as follows.

Algorithm 4.1 (CE for Continuous Optimization: Normal Updating)

1. **Initialize:** Choose $\hat{\boldsymbol{\mu}}_0$ and $\hat{\boldsymbol{\sigma}}_0^2$. Set $t = 1$.
2. **Draw:** Generate a random sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ from the $N(\hat{\boldsymbol{\mu}}_{t-1}, \hat{\boldsymbol{\sigma}}_{t-1}^2)$ distribution.
3. **Select:** Let \mathcal{I} be the indices of the N^e best performing (=elite) samples.
Update: For all $j = 1, \dots, n$ let

$$\tilde{\boldsymbol{\mu}}_{t,j} = \sum_{k \in \mathcal{I}} X_{k,j} / N^e, \quad (19)$$

$$\tilde{\boldsymbol{\sigma}}_{t,j}^2 = \sum_{k \in \mathcal{I}} (X_{k,j} - \tilde{\boldsymbol{\mu}}_{t,j})^2 / N^e. \quad (20)$$

4. Smooth:

$$\hat{\boldsymbol{\mu}}_t = \alpha \tilde{\boldsymbol{\mu}}_t + (1 - \alpha) \hat{\boldsymbol{\mu}}_{t-1}, \quad \hat{\boldsymbol{\sigma}}_t = \alpha \tilde{\boldsymbol{\sigma}}_t + (1 - \alpha) \hat{\boldsymbol{\sigma}}_{t-1}. \quad (21)$$

5. If $\max_j \{\hat{\sigma}_{t,j}\} < \epsilon$ **stop** and return μ_t (or the overall best solution generated by the algorithm) as the approximate solution to the optimization. Otherwise, increase t by 1 and return to Step 2.

For *constrained* continuous optimization problems, where the samples are restricted to a subset $\mathcal{X} \subset \mathbb{R}^n$, it is often possible to replace the normal sampling with sampling from a truncated normal distribution while retaining the updating formulas (19) and (20). An alternative is to use a beta distribution.

Smoothing, as in Step 4, is often crucial to prevent premature shrinking of the sampling distribution. Another approach is to *inject* extra variance into the sampling distribution, for example by increasing the components of σ^2 , once the distribution has degenerated; see the examples below and [Botev and Kroese \(2004\)](#).

4.1. Optimal control

Optimal control problems have been studied in many areas of science, engineering, and finance. Yet the numerical solution of such problems remains challenging. A number of different techniques have been used, including nonlinear and dynamic programming ([Bertsekas, 2007](#)), ant colony optimization ([Borzabadi and Mehne, 2009](#)), and genetic algorithms ([Wuerl et al., 2003](#)). For a comparison among these methods see [Borzabadi and Heidari \(2010\)](#). In this section we show how to apply the CE Algorithm 4.1 to optimal control problems over a fixed interval of time. As a particular example we consider the following nonlinear optimal control problem:

$$\begin{aligned} \min_{u \in \mathcal{U}} J\{u\} &= \min_{u \in \mathcal{U}} \int_0^1 \left(2z_1(\tau) - \frac{u(\tau)}{2} \right) d\tau, \\ \text{subject to : } &\quad \frac{dz_1}{d\tau} = u - z_1 + z_2 e^{-z_2^2/10}, \\ &\quad \frac{dz_2}{d\tau} = u - \pi^{5/2} z_1 \cos\left(\frac{\pi}{2}u\right), \\ &\quad z_1(0) = 0, \quad z_2(0) = 1, \quad |u(\tau)| \leq 1, \quad \tau \in [0, 1], \end{aligned}$$

where the function u belongs to the set \mathcal{U} of all piecewise continuous functions on $[0, 1]$. To obtain an approximate numerical solution we translate the infinite-dimensional *functional* optimization problem into a finite-dimensional *parametric* optimization problem. A simple way to achieve this is to divide the interval $[0, 1]$ into $n - 1$ subintervals $[\tau_1, \tau_2], \dots, [\tau_{n-1}, \tau_n]$ and approximate u via a spline that interpolates the points $\{(\tau_i, x_i)\}$, where $\mathbf{x} = (x_1, \dots, x_n)$ is a vector of *control points*. For a given vector \mathbf{x} of control points we can write the approximation as the linear combination:

$$u(\cdot) \approx u_{\mathbf{x}}(\cdot) = \sum_{k=1}^n x_k \phi_k(\cdot),$$

where $\{\phi_k \in \mathcal{U}\}$ are interpolating polynomials. For example, one of the simplest choices gives the *nearest neighbor* interpolation ($\tau_{n+1} = \tau_n$ and $\tau_0 = \tau_1$):

$$\phi_k(\tau) = \mathbb{I} \left\{ \frac{\tau_k + \tau_{k-1}}{2} < \tau < \frac{\tau_{k+1} + \tau_k}{2} \right\}. \quad (22)$$

The objective now is to find the optimal $u_x(\cdot)$ that solves the finite-dimensional control problem:

$$\begin{aligned} \min_{\mathbf{x}} S(\mathbf{x}) &= \min_{\mathbf{x}} \int_0^1 \left(2z_1(\tau) - \frac{u_x(\tau)}{2} \right) d\tau, \\ \text{subject to : } \quad \frac{dz_1}{d\tau} &= u_x - z_1 + z_2 e^{-z_2^2/10}, \\ \frac{dz_2}{d\tau} &= u_x - \pi^{5/2} z_1 \cos\left(\frac{\pi}{2} u_x\right), \\ z_1(0) = 0, \quad z_2(0) = 1, \quad |u_x(\tau)| &\leq 1, \quad \tau \in [0,1]. \end{aligned} \quad (23)$$

To find the optimal vector of control points $\mathbf{x} = (x_1, \dots, x_n)$ we apply Algorithm 4.1 in the following way.

1. In Step 2 we sample each component X_i of \mathbf{X} from a $N(\hat{\mu}_{t,j}, \hat{\sigma}_{t,j}^2)$ distribution, which is truncated to the interval $[-1, 1]$. This ensures that the approximation $u_x(\cdot)$ is constrained in the interval $[-1, 1]$.
2. We divide the time interval $[0, 1]$ into $n - 1$ equal subintervals and use (22) for the approximating curve. This gives an optimization problem of n dimensions.
3. For a given u_x we solve the nonlinear ODE system in (23) using the Runge–Kutta formulae of (Dormand et al., 1980).
4. The sample size is $N = 100$, the stopping parameter $\epsilon = 0.01$, and $\hat{\mu}_0 = \mathbf{0}$ and $\hat{\sigma}_0 = (10, \dots, 10)$, and $N^e = 20$. Since this is a minimization problem and $N^e = 20$, the elite samples are the first 20 control vectors that give the lowest score $S(\mathbf{x})$. We applied smoothing only to the parameter σ with $\alpha = 0.5$.

Figure 5 shows the typical evolution of the CE optimization Algorithm 4.1 with $n = 11$ control points. The figure shows the approximation $u_x(\cdot)$ and the control points $\mathbf{x} = \boldsymbol{\mu}_t$ (where $\boldsymbol{\mu}_t$ is the mean vector given in (19)) for $t = 1, 10, 20$, and the final iteration $t = 49$. For each graph we have recorded the corresponding value of J . In particular, for this simulation the smallest value found is $J\{u_x(\cdot)\} = 0.02691$. Observe the qualitative behavior of the optimal control: u starts near 1 and gradually decreases to $u = -1$ until at about $\tau = 0.7$ the control abruptly switches to $u = 1$ and remains so till the final time $\tau = 1$.

Figure 6 shows the evolution of the standard deviation vector $\hat{\sigma}_t$ associated with the sampling distribution $N(\hat{\mu}_t, \hat{\sigma}_t^2)$ of each random control vector \mathbf{X} . Notice that all of the components of $\hat{\sigma}_t$ shrink to zero as the iterations progress, and that since $\hat{\sigma}_{t,7}$ and $\hat{\sigma}_{t,8}$ are the last to decay, the control points x_6 and x_7 are the last to converge.

It is desirable to investigate the dependence of the solution on the number of control points. Note that while increasing the number of control points \mathbf{x} improves the approximation of u , it also makes the optimization problem more difficult, because of the increased dimensions of the optimization problem. Figure 7 shows the evolution of the CE optimization Algorithm 4.1 with $n = 21$ control points, where the interval $[0, 1]$ is again subdivided into equal segments. The algorithmic setup and

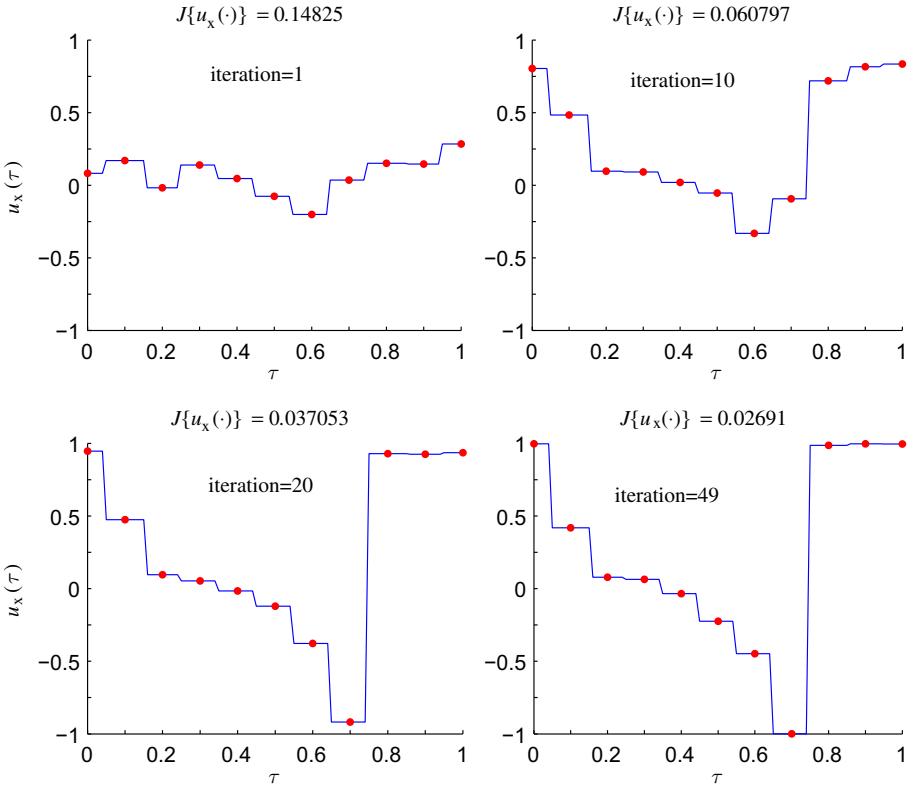


Fig. 5. The evolution of the CE optimization method on the nonlinear optimal control problem with $n = 11$ control points. The graphs show the control $u_{\mu_t}(\cdot)$ for $t = 1, 10, 20$, and the final $t = 49$. The control points μ_t are shown as dots on the curve $u_{\mu_t}(\cdot)$.

the values of all parameters are kept the same. In this case the optimization effort was larger and the algorithm terminated in 88 iterations. Observe that as a result of using more control points we were able to achieve a lower minimum for J , namely, $J\{u_x(\cdot)\} = 0.018395$.

We were not able to obtain a lower value for J using more than 21 points due to the increased complexity of the optimization. In addition, smooth cubic splines did not yield better solutions than the simple nearest neighbor interpolation due to the discontinuous nature of the optimal control.

For more examples of using the CE method for optimal control problems see Sani and Kroese (2008, 2010). In addition to a number of simple examples, Sani and Kroese (2008) apply the CE method to the optimization of a large-scale model for the spread of HIV/AIDS.

4.2. Maximum likelihood optimization

Mixture models are widely used for clustering and pattern recognition (McLachlan and Peel, 2000). A finite mixture pdf is a density of the form

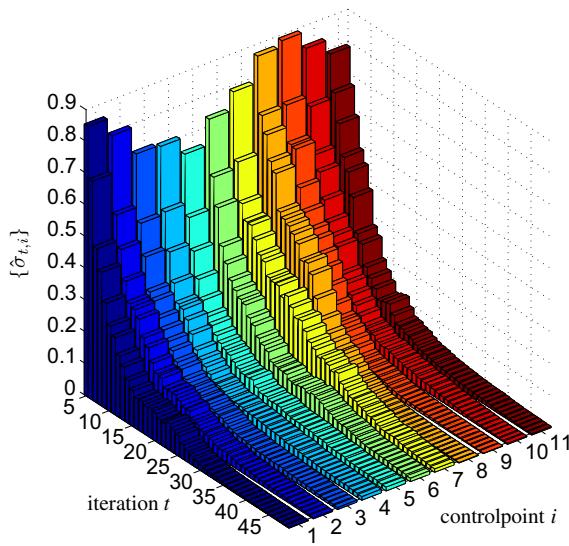


Fig. 6. Evolution of the vector $\hat{\sigma}_t$ for iterations $t = 5, \dots, 49$.

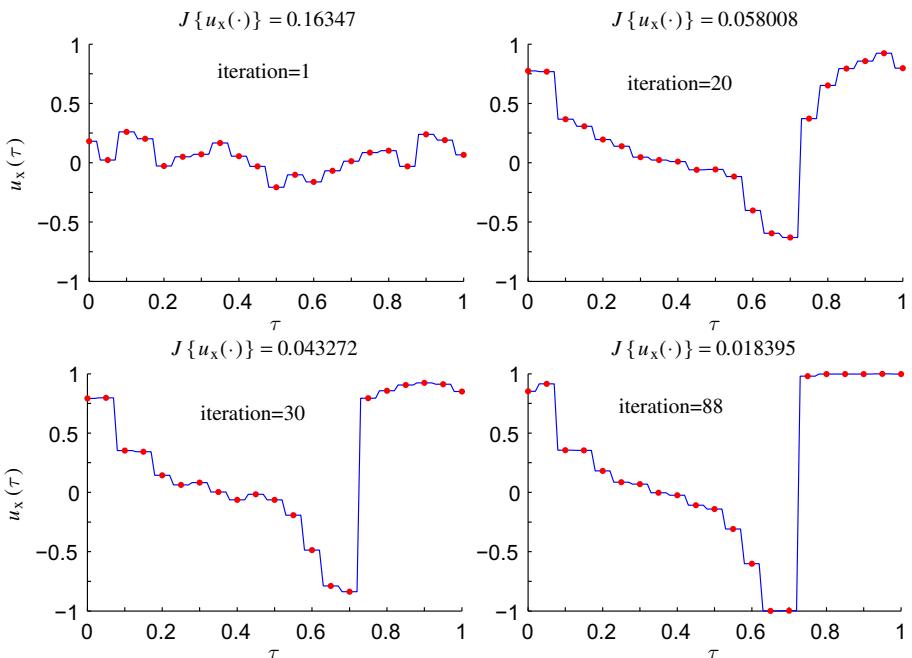


Fig. 7. The evolution of the CE optimization method on the nonlinear optimal control problem with $n = 21$ control points. The optimization terminated in 88 iterations and achieved the minimum of $J\{u_x(\cdot)\} = 0.018395$.

$$f(\mathbf{y}) = \sum_{k=1}^c w_k f_k(\mathbf{y}),$$

where $\{w_k\}$ are probabilities summing to 1, $\{f_k\}$ are pdfs, and c is the number of mixture components. The pdfs $\{f_k\}$ may or may not belong to a single parametric family. The fitting of mixture models is best carried out using the principle of maximum likelihood and requires the maximization of a likelihood function with many local maxima and saddle points. Typically, the EM algorithm of Dempster et al. (1977) is the most efficient method for fitting of mixture models when $\{f_k\}$ belong to the exponential family of parametric pdfs. However, the EM algorithm is either inapplicable or inefficient when the densities $\{f_k\}$ are not part of an exponential family or are not continuous with respect to some of its model parameters. For example, consider a mixture consisting of one d -dimensional Gaussian and $c - 1$ uniform densities on d -dimensional rectangles:

$$\begin{aligned} f(\mathbf{y}; \boldsymbol{\theta}) &= \frac{w_1 e^{-\frac{1}{2} \sum_{i=1}^d (y_i - v_{1,i})^2 / \varsigma_{1,i}^2}}{(2\pi)^{\frac{d}{2}} \prod_{i=1}^d \varsigma_{1,i}} \\ &\quad + \sum_{k=2}^c w_k \prod_{i=1}^d \frac{I\{|y_i - v_{k,i}| < \frac{1}{2} \varsigma_{k,i}\}}{\varsigma_{k,i}}. \end{aligned} \quad (24)$$

Here, $(v_{k,1}, \dots, v_{k,d})$ and $(\varsigma_{k,1}, \dots, \varsigma_{k,d})$ are the location and scale parameters of the k th component, and

$$\boldsymbol{\theta} = ((v_{1,1}, \dots, v_{c,d}), (\varsigma_{1,1}, \dots, \varsigma_{c,d}), (w_1, \dots, w_c))$$

summarizes all model parameters. The log-likelihood function given the data $\mathbf{y}_1, \dots, \mathbf{y}_n$ is $S(\boldsymbol{\theta}) = \sum_{j=1}^n \ln f(\mathbf{y}_j; \boldsymbol{\theta})$, which is not continuous with respect to all parameters, because the support of each uniform component depends on the scale parameters $\{\varsigma_{k,i}, k \geq 2\}$. We now show how to apply the CE method to solve the optimization program $\hat{\boldsymbol{\theta}} = \operatorname{argmax}_{\boldsymbol{\theta} \in \Theta} S(\boldsymbol{\theta})$, where Θ is the set for which all $\varsigma_{k,i} > \varsigma_{\text{low}} > 0$ for some threshold ς_{low} and $\{w_k\}$ are probabilities summing to one. Note that ς_{low} must be strictly positive to ensure that $\max_{\boldsymbol{\theta} \in \Theta} S(\boldsymbol{\theta}) < \infty$.

In Step 2 of Algorithm 4.1 the population $\mathbf{X}_1, \dots, \mathbf{X}_N$ corresponds to randomly generated proposals for the parameter $\boldsymbol{\theta} = \mathbf{X}$. Thus, the CE parameter vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are of length $(2d + 1)c$, with the first dc elements associated with the location parameters of the mixture, the next dc elements associated with the scale parameters, and the last c elements associated with the mixture weights. In particular, we sample the locations $v_{1,1}, \dots, v_{c,d}$ from the normal distributions $N(\mu_1, \sigma_1^2), \dots, N(\mu_{cd}, \sigma_{cd}^2)$. To account for the constraints on the scale parameters of the mixture we sample $\varsigma_{1,1}, \dots, \varsigma_{c,d}$ from the normal distributions $N(\mu_{cd+1}, \sigma_{cd+1}^2), \dots, N(\mu_{2cd}, \sigma_{2cd}^2)$ truncated to the set $[\varsigma_{\text{low}}, \infty)$. To generate the mixture weights $\{w_k\}$ we need to sample positive random variables $\omega_1, \dots, \omega_c$ such that $\sum_k \omega_k = 1$. For this purpose we generate all weights from truncated normal distributions in a random order (specified by a random permutation), as in the following algorithm.

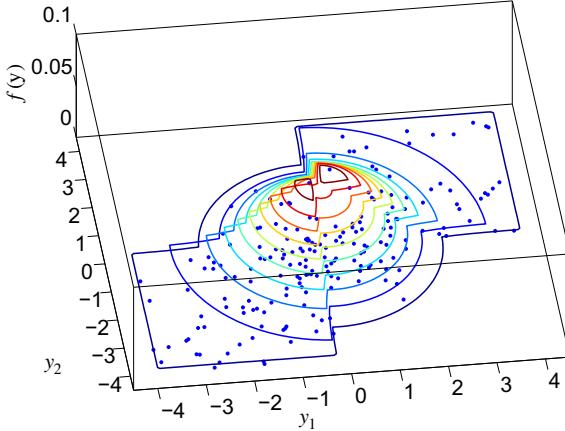


Fig. 8. A sample of size 200 from a mixture of two uniform and a Gaussian distributions. Superimposed is the contour plot of the fitted mixture density $f(\mathbf{y}; \hat{\boldsymbol{\theta}})$.

Algorithm 4.2 (Generation of Mixture Weights)

Require: The last c elements of the CE parameter vectors μ and σ^2 .

Generate a random permutation π of $1, \dots, c$.

for $k = 1, \dots, (c - 1)$ **do**

 Generate $w_{\pi(k)}$ from the normal distribution $N(\mu_{2dc+\pi(k)}, \sigma_{2dc+\pi(k)}^2)$ truncated to the interval $[0, 1 - (w_{\pi(1)} + \dots + w_{\pi(k-1)})]$.

end for

Set $w_{\pi(c)} = 1 - (w_{\pi(1)} + \dots + w_{\pi(c-1)})$.

return Random mixture weights w_1, \dots, w_c .

As a particular example, consider fitting the mixture model (24) to the $n = 200$ random points depicted on Fig. 8. The data are pseudo-random variables generated from (24) with $c = 3$, weights $(w_1, w_2, w_3) = (1/2, 1/4, 1/4)$, location parameters

$$\begin{pmatrix} \nu_{1,1} \\ \nu_{1,2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \nu_{2,1} \\ \nu_{2,2} \end{pmatrix} = \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \begin{pmatrix} \nu_{3,1} \\ \nu_{3,2} \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix},$$

and scale parameters

$$\begin{pmatrix} \varsigma_{1,1} \\ \varsigma_{1,2} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} \varsigma_{2,1} \\ \varsigma_{2,2} \end{pmatrix} = \begin{pmatrix} \varsigma_{3,1} \\ \varsigma_{3,2} \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix}.$$

We applied Algorithm 4.1 with $N = 100, N^e = 10, \alpha = 0.5, \epsilon = \varsigma_{\text{low}} = 0.01$, and $\hat{\mu}_0 = (1, \dots, 1)$ and $\hat{\sigma}_0 = (10, \dots, 10)$. The algorithm terminated after $t = 89$ iterations with the overall maximum log-likelihood of $S(\hat{\boldsymbol{\theta}}) = -707.7593$. Figure 8 shows the estimated mixture model. The model appears to fit the data well. In addition, the true value of the parameters $\boldsymbol{\theta}$ gives $S(\boldsymbol{\theta}) = -710.7937 < S(\hat{\boldsymbol{\theta}})$, which suggests that $\hat{\boldsymbol{\theta}}$ is the global maximizer. We found that for fitting mixture models the CE algorithm is not sensitive to the initial conditions and works satisfactorily for $0.2 \leq \alpha \leq 0.8$.

5. Summary

We have described the CE method for combinatorial and continuous optimization. In summary, any CE algorithm for optimization involves the following two main iterative phases:

1. *Generate*: a random sample of objects in the search space \mathcal{X} (trajectories, vectors, etc.) according to a specified probability distribution.
2. *Update*: or refit the parameters of that distribution based on the N^e best performing samples (the so-called elite samples) using CE minimization.

An important part of the algorithm involves the selection of a suitable probability density that allows for simple random variable generation and simple updating formulae (arising from the CE minimization) for the parameters of the density. For most optimization problems standard parametric models such as the multivariate Bernoulli and Gaussian densities (for discrete and continuous problems, respectively) are adequate. Finally, the CE method is a robust tool for noisy continuous or discrete optimization problems.

References

- Alon, G., Kroese, D.P., Raviv, T., Rubinstein, R.Y., 2005. Application of the cross-entropy method to the buffer allocation problem in a simulation-based environment. *Ann. Oper. Res.* 134 (1), 137–151.
- Bertsekas, D.P., 2007. *Dynamic Programming and Optimal Control*, third ed. vol. I. Athena Scientific.
- Borzabadi, A.H., Heidari, M., 2010. Comparison of some evolutionary algorithms for approximate solutions of optimal control problems. *Aust. J. Basic Appl. Sci.* 4 (8), 3366–3382.
- Borzabadi, A.H., Mehne, H.H., 2009. Ant colony optimization for optimal control problems. *J. Inform. Comput. Sci.* 4 (4), 259–264.
- Botev, Z.I., Kroese, D.P., 2004. Global likelihood optimization via the cross-entropy method with an application to mixture models. In: Proceedings of the 36th Winter Simulation Conference, Washington, D.C., pp. 529–535.
- Boubezoula, A., Paris, S., Ouladsinea, M., 2008. Application of the cross entropy method to the GLVQ algorithm. *Pattern Recog.* 41 (10), 3173–3178.
- Busoniu, L., Babuska, R., De Schutter, B., Ernst, D., 2010. *Reinforcement Learning and Dynamic Programming using Function Approximators*. Taylor & Francis Group, New York.
- Cancela, H., Urquhart, M.E., 1995. Simulated annealing for communication network reliability improvements. In: Proceedings of the XXI Latin American Conference on Informatics, pp. 1413–1424.
- Chepuri, K., Homem-de-Mello, T., 2005. Solving the vehicle routing problem with stochastic demands using the cross entropy method. *Ann. Oper. Res.* 134 (1), 153–181.
- Cohen, I., Golany, B., Shtub, A., 2007. Resource allocation in stochastic, finite-capacity, multi-project systems through the cross entropy methodology. *J. Scheduling* 10 (1), 181–193.
- Costa, A., Owen, J., Kroese, D.P., 2007. Convergence properties of the cross-entropy method for discrete optimization. *Oper. Res. Lett.* 35 (5), 573–580.
- de Boer, P.T., Kroese, D.P., Mannor, S., Rubinstein, R.Y., 2005. A tutorial on the cross-entropy method. *Ann. Oper. Res.* 134 (1), 19–67.
- Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc. Ser. B* 39 (1), 1–38.
- Dengiz, B., Altiparmak, F., Smith, A.E., 1997. Local search genetic algorithm for optimal design of reliable networks. *IEEE Trans. Evol. Comput.* 1 (3), 179–188.
- Dormand, J.R., Prince, P.J., 1980. A family of embedded Runge-Kutta formulae. *J. Comp. Appl. Math.* 6, 19–26.

- Elperin, T., Gertsbakh, I.B., Lomonosov, M., 1991. Estimation of network reliability using graph evolution models. *IEEE Trans. Reliability* 40 (5), 572–581.
- Ernst, D., Glavic, M., Stan, G.-B., Mannor, S., Wehenkel, L., Supelec, Gif sur Yvette, 2007. The cross-entropy method for power system combinatorial optimization problems. In: *IEEE Lausanne, Power Tech*, 2007 pp. 1290–1295.
- Evans, G.E., Keith, J.M., Kroese, D.P., 2007. Parallel cross-entropy optimization. In: *Proceedings of the 2007 Winter Simulation Conference*, Washington, D.C., pp. 2196–2202.
- Gu, J., Purdom, P.W., Franco J., Wah, B.W., 1997. Algorithms for the satisfiability (SAT) problem: a survey. In: *Satisfiability Problem: Theory and Applications*. American Mathematical Society, Providence, RI, pp. 19–152.
- Hintsanen, P., Toivonen, H., Sevon, P., 2010. Fast discovery of reliable subnetworks. In: *International Conference on Advances in Social Network Analysis and Mining 2010*, pp. 104–111.
- Hoos, H.H., Stützle, T., 2000. SATLIB: an online resource for research on SAT. In: Gent, I.P., Maaren, H.v., Walsh, T., (Eds.), *SAT 2000*. IOS Press, pp. 283–292 <www.satlib.org>.
- Keith, J., Kroese, D.P., 2002. Sequence alignment by rare event simulation. In: *Proceedings of the 2002 Winter Simulation Conference*, San Diego, pp. 320–327.
- Kellerer, H., Pferschy, U., Pisinger, D., 2004. *Knapsack Problems*, first ed. Springer Verlag.
- Kothari, R.P., Kroese, D.P., 2009. Optimal generation expansion planning via the cross-entropy method. In: Rossetti, M.D., Hill, R.R., Johansson, B., Dunkin, A., Ingalls, R.G., (Eds.), *Proceedings of the 2009 Winter Simulation Conference*, pp. 1482–1491.
- Kroese, D.P., Porotsky, S., Rubinstein, R.Y., 2006. The cross-entropy method for continuous multi-extremal optimization. *Methodol. Comput. Appl. Probab.* 8 (3), 383–407.
- Kroese, D.P., Hui, K.-P., Narai, S., 2007a. Network reliability optimization via the cross-entropy method. *IEEE Trans. Reliability* 56 (2), 275–287.
- Kroese, D.P., Rubinstein, R.Y., Taimre, T., 2007b. Application of the cross-entropy method to clustering and vector quantization. *J. Global Optim.* 37, 137–157.
- Leite de Silva, A.M., Fernandez, R.A.G., Singh, C., 2010. Generating capacity reliability evaluation based on Monte Carlo simulation and cross-entropy methods. *IEEE Trans. Power Syst.* 25 (1), 129–137.
- Liu, Z., Doucet, A., Singh, S.S., 2004. The cross-entropy method for blind multiuser detection. In: *IEEE International Symposium on Information Theory*, Chicago, Piscataway.
- Lörincza, A., Palotaia, Z., Szirtesb, G., 2008. Spike-based cross-entropy method for reconstruction. *Neurocomputing* 71 (16–18), 3635–3639.
- Margolin, L., 2005. On the convergence of the cross-entropy method. *Ann. Oper. Res.* 134 (1), 201–214.
- McLachlan, G.J., Peel, D., 2000. *Finite Mixture Models*. John Wiley & Sons, New York.
- Menache, I., Mannor, S., Shimkin, N., 2005. Basis function adaptation in temporal difference reinforcement learning. *Ann. Oper. Res.* 134 (1), 215–238.
- Pihur, V., Datta, S., Datta, S., 2007. Weighted rank aggregation of cluster validation measures: a Monte Carlo cross-entropy approach. *Bioinformatics* 23 (13), 1607–1615.
- Reichelt, D., Rothlauf, F., Bmilkowsky, P., 2007. Designing reliable communication networks with a genetic algorithm using a repair heuristic. In: *Evolutionary Computation in Combinatorial Optimization*. Springer-Verlag, Heidelberg, pp. 177–186.
- Rubino, G., 1998. Network reliability evaluation. In: Walrand, J., Bagchi, K., Zobrist, G.W. (Eds.), *Network Performance Modeling and Simulation*. Blackwell Scientific Publications, Amsterdam (Chapter 11).
- Rubino, G., Tuffin, B., 2009. *Rare Event Simulation*. John Wiley & Sons, New York.
- Rubinstein, R.Y., 1997. Optimization of computer simulation models with rare events. *Eur. J. Oper. Res.* 99 (1), 89–112.
- Rubinstein, R.Y., 1999. The cross-entropy method for combinatorial and continuous optimization. *Method. Comput. Appl. Prob.* 1 (2), 127–190.
- Rubinstein, R.Y., 2001. Combinatorial optimization, cross-entropy, ants and rare events. In: Uryasev, S., Pardalos, P.M. (Eds.), *Stochastic Optimization: Algorithms and Applications*. Kluwer, Dordrecht, pp. 304–358.
- Rubinstein, R.Y., Kroese, D.P., 2004. The cross-entropy method: a unified approach to combinatorial optimization. *Monte Carlo Simulation and Machine Learning*. Springer-Verlag, New York.
- Rubinstein, R.Y., Kroese, D.P., 2007. *Simulation and the Monte Carlo Method*, second ed. John Wiley & Sons, New York.

- Sani, A., 2009. Stochastic modelling and intervention of the spread of HIV/AIDS. Ph.D. Thesis, The University of Queensland, Brisbane.
- Sani, A., 2010. The Spread of HIV/AIDS in Mobile Populations: Modelling, Analysis, Simulation. LAP LAMBERT Academic Publishing, Berlin.
- Sani, A., Kroese, D.P., 2008. Controlling the number of HIV infectives in a mobile population. *Math. Biosci.* 213 (2), 103–112.
- Senju, S., Toyoda, Y., 1968. An approach to linear programming with 0–1 variables. *Manag. Sci.* 15 (4), B196–B207.
- Szabó, Z., Póczos, B., Lörincz, A., 2006. Cross-entropy optimization for independent process analysis. In: Independent Component Analysis and Blind Signal Separation, vol. 3889. Springer-Verlag, Heidelberg, pp. 909–916.
- Unveren, A., Acan, A., 2007. Multi-objective optimization with cross entropy method: stochastic learning with clustered pareto fronts. In: IEEE Congress on Evolutionary Computation (CEC 2007), pp. 3065–3071.
- Wang, G.-B., Huang, H.-Z., Liu, Y., Zhang, X., Wang, Z., 2009. Uncertainty estimation of reliability redundancy in complex systems based on the cross-entropy method. *J. Mech. Sci. Technol.* 23, 2612–2623.
- Won, J.-M., Karray, F., 2010. Cumulative update of all-terminal reliability for faster feasibility decision. *IEEE Trans. Reliability* 59 (3), 551–562.
- Wu, Y., Fyfe, C., 2008. Topology preserving mappings using cross entropy adaptation. In: 7th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases, Cambridge.
- Wuerl, A., Crain, T., Braden, E., 2003. Genetic algorithm and calculus of variations-based trajectory optimization technique. *J. Spacecraft Rockets* 4 (6), 882–888.

This page is intentionally left blank

Probability Collectives in Optimization

*David H. Wolpert¹, Stefan R. Bieniawski²,
and Dev G. Rajnarayan³*

¹*The Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, NM 87501, USA*

²*Boeing Research and Technology, P.O. Box 3707 MC 42-51, Seattle,
WA 98124-2207, USA*

³*Desktop Aeronautics, Inc., 1900 Embarcadero Road, Suite 101, Palo Alto,
CA 94303, USA*

Abstract

This article concerns “blackbox optimization” algorithms in which one iterates the following procedure: Choose a value $x \in X$, getting statistical information about an associated value $G(x)$, then use the set of all pairs $\{(x, G(x))\}$ found so far to choose a next x value at which to sample G , the goal being to find x s with as small $G(x)$ as possible, and to do so as fast as possible. Examples of conventional blackbox optimization algorithms are genetic algorithms, simulated annealing, etc. These conventional algorithms work directly with values x , stochastically mapping the set $\{(x, G(x))\}$ to the next x . The distribution over new x s that gets sampled is never explicitly optimized. In contrast, in the Probability Collectives (PC) approach, one explicitly uses the set $\{(x, G(x))\}$ to optimize the probability distribution over x that will be sampled. This article reviews some of the work that has been done on Probability Collectives, in particular presenting some of the many experiments that have demonstrated its power.

Keywords: optimization, machine learning, probability collectives, cross-validation, stacking

1. Introduction

1.1. Intuition behind Probability Collectives

This article concerns “blackbox optimization” algorithms. In these algorithms one iterates a process of choosing a value $x \in X$, getting statistical information about an associated value $G(x)$, and then using all pairs $\{(x, G(x))\}$ found so far to choose a next x value at which to sample G . The goal is to find x s with as small $G(x)$ as possible. In the versions of blackbox optimization we focus on here, at no point in

this iterative process does one have any knowledge concerning G beyond the set of sample pairs found so far.

Conventional approaches to blackbox optimization work directly with values x , in that they specify a map K from any set of pairs $\{(x, G(x))\}$ to a next sample x . In the “Probability Collectives” (PC) approach, one instead works with probability distributions over x , in that one specifies a map κ from any set of pairs $\{(x, G(x))\}$ to a next *distribution* over X , $q(x)$, which gets sampled to form a next x .¹ In conventional approaches to blackbox optimization one tries to design K so that after iterating it one is likely to produce x s with small values of $G(x)$. In PC one instead tries to design κ so that after iterating it one is likely to produce $q(x)$ s that are peaked about small values of $G(x)$. There are many ways to formalize this goal, e.g., as finding the q that minimizes the expected value $\int dx G(x)q(x)$. Indeed, the optimal q is one whose support is restricted to the minimizers of $G(x)$, so that sampling those q s will give x s that minimize $G(x)$.

There are many advantages to working with distributions over X , as in PC, rather than working with x values directly. For example, many PC algorithms do not need to be modified even if one changes the type of the space X , so long as X has some very general characteristics (e.g., is finite). This is because whatever the type of X , a probability distribution over X —which is what the PC algorithm works on—is the same kind of object, namely a vector of real numbers. Indeed, PC’s goal of iteratively modifying the distribution q to find low values of $\int dx G(x)q(x)$ amounts to iteratively modifying a real-valued vector q to minimize a real-valued function of that vector. This is exactly what is done in optimization schemes like gradient descent or Newton’s method. Accordingly PC can leverage the power of such schemes. In particular, we can do this even if X is a categorical, finite space. So by using PC, “gradient descent for categorical variables” is perfectly well defined.

Another advantage is that often the q produced in a PC-based approach will be tightly peaked in certain dimensions, while being broad in other dimensions. This provides sensitivity information concerning the relative importance to solving the optimization problem of getting the values of those dimensions precisely correct. Another advantage we get by optimizing q is that we can initialize it to a set of broad peaks each centered on a solution x^l produced by some other optimization algorithm. Then as that initial q gets updated, the set of solutions provided by those other optimization algorithms are in essence combined, to produce a solution that should be superior to any of them individually.

1.2. Delayed and immediate sampling PC

There are two basic types of PC that have been explored to date, *delayed sampling* and *immediate sampling*. Historically, delayed sampling PC was investigated first. It was originally motivated by restricting q to product distributions, and casting the goal of the PC algorithm as finding such a q that minimizes $\int dx G(x)q(x)$. For this setup one

¹ Throughout this paper, the measure over X is implicit, and we will loosely use terms like “probability distribution” even if X is uncountable and we should really refer to “probability density functions.” Similar we will typically refer to “integrals” even when there is a point mass measure, so that we should really refer to “sums.”

can derive simple algebraic formulas for how q should be updated from one iteration to the next, if one knew certain expectation values perfectly. Since in practice one does not know those expectation values perfectly, as a final step in the derivation of the update rule one has to introduce Monte Carlo (MC) techniques. (The name derives from the fact that the MC arises after the algebraic manipulations.)

Using product distributions has the advantage that it lends itself to parallel implementation (the separate component distributions of the product can be allocated to the separate parallel systems). That make is particularly well-suited to distributed control applications. However product distributions have the disadvantage that they sometimes result in slow convergence of the associated PC algorithm, e.g., when the value that is best for one component of x depends critically on the value for another component of x . Although there are some tricks for circumventing this problem (e.g., the “semi-coordinate transformations” described in [Wolpert et al. \(2006\)](#)), it is often advantageous to avoid the problem entirely and use distributions that are not product distributions.

Allowing such arbitrary classes of probability distributions is the primary motivation for immediate sampling PC. We write the goal of immediate sampling PC as finding the parameter θ that minimizes an integral transform

$$\mathcal{F}_{\mathcal{G}}(q_{\theta}) \triangleq \int dx dg P(g|x, \mathcal{G}) F(g, q_{\theta}(x)) \quad (1)$$

for a distribution q_{θ} in some class of θ -parameterized distributions. To motivate this notation, let \mathcal{G} specify a “blackbox oracle” that stochastically returns real values g in response to any input x . Also take $F(g, q_{\theta}(x)) = gq_{\theta}(x)$. Then $\mathcal{F}_{\mathcal{G}}(q_{\theta})$ is the average value of $g(x)$, evaluated under the x -distribution $q_{\theta}(x)$, with the g value at x given stochastically by $P(g|x, \mathcal{G})$. In particular, if $P(g|x, \mathcal{G}) = \delta(g, G(x))$ for some function G (i.e., if g is a deterministic function of x), and θ runs over the space of product distributions, then $F(g, q_{\theta}(x))$ is just the quantity we wish to minimize in delayed sampling PC. However the notation (and techniques) of immediate sampling PC applies more broadly.

In immediate sampling PC we use MC techniques “immediately,” to estimate $\mathcal{F}_{\mathcal{G}}(q_{\theta})$ from the given set of samples for all values of θ . We then use powerful techniques from machine learning to choose among the possible θ s based on those associated estimates. The algebraic manipulations used in delayed sampling before any MC estimation is done are absent in immediate sampling, since those manipulations are only possible for the special case where q_{θ} is the set of all product distributions.

1.3. Relevant literature and roadmap of this paper

In [Wolpert \(2004a\)](#) can be found pedagogical examples where there are closed-form solutions for the q produced by an elementary delayed sampling PC optimization algorithm. Also presented there is a proof that in the infinitesimal limit, many techniques for updating q become identical; these techniques all become a variant of Evolutionary Game theory’s replicator dynamics in that limit. See [Wolpert and Bieciawski \(2004a\)](#) for an explicit formulation of how to apply delayed sampling PC to scenarios where the underlying variable is the trajectory of a multi-dimensional variable through time, i.e., to a policy-optimization scenario. Related connections

between game theory, statistical physics, information theory, and PC are discussed in [Wolpert \(2004b\)](#).

See [Bieniawski et al. \(2004, 2005\)](#), [Macready et al. \(2004\)](#), [Fan Lee and Wolpert \(2004\)](#), [Wolpert and Bieniawski \(2004a,b\)](#), and [Antoine et al. \(2004\)](#) for other work on delayed sampling PC, including both software and hardware experiments, for both optimization and control. In particular, see [Wolpert and Lee \(2004\)](#), [Wolpert \(2004\)](#), and [Wolpert and Bieniawski \(2004\)](#) for work showing, respectively, how to use delayed sampling PC to improve Metropolis-Hastings sampling, how to relate delayed sampling PC to the mechanism design work in [Wolpert et al. \(1999\)](#), [Wolpert and Tumer \(2001, 2002\)](#), [Wolpert \(2003\)](#), and how to extend it to continuous move spaces and time-extended strategies.

There are other probability-based approaches to optimization and control, including ([Megiddo, 1976](#); [Fudenberg and Levine, 1998](#); [Shamma and Arslan, 2004](#); [Jaynes, 1957](#); [De Bonet, 1997](#); [Rubinstein and Kroese, 2004](#); [Sabes and Jordan, 1995](#)). See ([Wolpert, 2004b](#)) for a discussion that relates delayed sampling PC (without using that name) to many of these. See [Wolpert et al. \(2004\)](#) and [Wolpert and Tumer \(2001\)](#) for earlier, less formal work related to delayed sampling PC.

Finally, see [Wolpert et al. \(2006\)](#) for an overview of delayed sampling PC, and for many extensions of the basic delayed sampling PC algorithms, including extensions for the case where we have constraints on X , and extensions that overcome the restrictions imposed by product distributions, while still retaining the distributed nature of the algorithm.

Since immediate sampling is a more recent body of techniques, fewer papers have been produced so far on it. See [Rajnarayan et al. \(2011, 2007, 2008\)](#) and [Wolpert and Rajnarayan \(2007\)](#).

In Section 2 we present a quick summary of the theory of delayed sampling PC. Section 3 then presents some implementation details and experiments. In Section 4 we present a quick summary of the theory of immediate sampling PC. Section 5 then presents some implementation details and experiments.

2. Delayed sampling theory

2.1. The Maxent Lagrangian

Say we are given the following problem:

(P3): Find

$$\min_{\{q_i\}} \int dx G(x) \prod_i q_i(x_i)$$

such that

$$\int dx_i q_i(x_i) = 1 \quad \forall i$$

$$q_i(x_i) \geq 0 \quad \forall i, x_i,$$

where each q_i is a single-dimensional real variable, which we will sometimes refer to as the “move” or “choice” of “agent i .”

In conventional continuous optimization one standard way to solve (P3) is by replacing the inequality constraints with *barrier functions* $\phi_i : \mathbb{R} \rightarrow \mathbb{R}$ each of which is nowhere negative and which is also infinite for negative values of its argument (Boyd and Vandenberghe, 2003). This replaces the inequality constrained problem (P3) with the following problem having no inequality constraints:

(P1): Find

$$\min_{\{q_i\}} \left[\int dx G(x) \prod_i q_i(x_i) + \sum_{i=1}^N \int dx_i \mu(i, x_i) \phi_i(q_i(x_i)) \right],$$

such that

$$\int dx_i q_i(x_i) = 1 \quad \forall i,$$

where the non-negative real values $\mu(i, x_i)$ are the *barrier parameters*. Due to the nature of the barrier functions, we know that the solution to (P1) must obey the inequality constraints for any allowed values of the barrier parameters. Enforcing the equality constraints of (P1) as well guarantees that out q meets all of our constraints, i.e., is feasible.

Typically with barrier function methods one solves a sequenced of optimization problems, each of which has fixed barrier parameters. Each such problem starts with the q produced by the solution to the preceding problem. The difference between two successive problems is that the barrier parameter has been slightly reduced. As such an annealing progresses the barrier parameters disappear and (P1) becomes (P3), i.e., our final q is both feasible and (at least locally) solves the minimization over q of problem (P3). For convex objective functions, this process has certain guarantees of converging to the global optimum of (P3). (Note though that our objective function, $\mathbb{E}_q(G)$, is not a convex function of the components of q ; due to the fact that q is a product distribution, $\mathbb{E}_q(G) = \int dx \prod_i q_i(x_i) G(x)$ is a multinomial function of q .)

One common choice of barrier function is $\phi(y) = y \ln(y)$ for $y > 0$, $\phi(y) = \infty$ otherwise.² Say we also take $\mu(i, x_i)$ to be independent of x_i , so we can write it as the vector with components $\{T_i\}$. For this case, when q is normalized (i.e., satisfies the equality constraints), the operand of the min operator becomes $\mathbb{E}_q(G) - \sum_i T_i S(q_i)$, where $S(q)$ is the Shannon entropy of q_i . If T_i is independent of i , this difference becomes $\mathbb{E}_q(G) - TS(q)$. This expression is called the *free energy* in statistical physics, if we identify G with the “Hamiltonian” of the system and T with its “temperature.” It is also the *Kullbach-Leibler* distance from q to the Boltzmann distribution $p(x) \propto \exp(-G(x)/T)$, up to an irrelevant q -independent additive constant.³ See Wolpert and Biegniewski (2004b) for a discussion of the shape of the function taking q to qp KL distance for product distributions q .

²Formally, for ϕ to be a barrier function, we must add a constant large enough so that ϕ is nowhere negative, but such a constant is irrelevant for our purposes.

³Recall that the Kullbach-Leibler distance from an arbitrary distribution $q(x)$ to an arbitrary distribution $p(x)$ is $-\int dx q(x) \ln[p(x)/q(x)]$. We will sometimes call this the *qp KL distance*, to distinguish it from the *pq KL distance* which is the Kullbach Leibler distance from p to q . These two distances are also sometimes called the “M-projection” and the “I projection,” respectively, in the literature. In general, KL distance between two distributions is non-negative, and equals 0 iff those distributions are identical. See Cover and Thomas (1991), Mackay (2003), and Koller and Friedman (2009).

Continuing with our choice of entropic barrier function, we can solve (P1) via Lagrange parameters in the usual way, getting the Lagrangian

$$\mathcal{L}(q, \vec{T}) = \mathbb{E}_q(G) - \sum_i T_i S(q_i) + \sum_i \lambda_i \left[\int dx_i q_i(x_i) - 1 \right] \quad (2)$$

with the obvious extension of the definition of E and S to all of the space of real-valued possible functions over x . For simplicity from now on we will take all the T_i to be the same. In this case we refer to the expression in Eq. (2), as the *Maxent Lagrangian*, since it can also be justified using the Maximum entropy principle of information theory (Wolpert, 2004b). It is just the free energy plus the dot product of the Lagrange parameter vector with the equality constraint functions. (Many other Lagrangians have also been explored; see some of the literature mentioned above.)

2.2. The gradient of the free energy

Say that T is fixed. In that case the entropy term in the free energy is globally convex in the remaining free variable, q . The $\mathbb{E}_q(G)$ term in the free energy would also be convex, if we had only a single agent (in that case $\mathbb{E}_q(G)$ would be linear in q). Accordingly, the free energy would be convex over \mathcal{Q} , the space of all product distributions over X . That is also true of the equality constraints, no matter how many agents we have. So the full Maxent Lagrangian would be convex if we had only a single agent, and the usual associated guarantees about duality gaps, saddle point solutions, etc., would apply. However for multiple agents $\mathbb{E}_q(G)$ is not linear in q , and we do not have those convexity guarantees.

For this more general case, the simplest thing to do is an iterative descent of the free energy over \mathcal{Q} , where at every iteration we step in the direction within \mathcal{Q} that, to first order, maximizes the drop in free energy. Since the step is restricted to lie within \mathcal{Q} , the equality constraints of the Maxent Lagrangian are obeyed automatically. Expanding to first order the step that maximizes the drop in \mathcal{L} is given by (-1 times)

$$\nabla_q \mathcal{L}(q) - \eta(q), \quad (3)$$

where the $\eta(q)$ term is chosen to ensure that we stay on \mathcal{Q} .⁴ To evaluate the $q_i(x_i)$ component of the gradient (one such component for every agent i and every possible move x_i by that agent) we write

$$[\nabla_q \mathcal{L}(q)]_{q_i(x_i)} = \frac{\partial \mathcal{L}}{\partial q_i(x_i)} = \mathbb{E}_{q_{-i}}(G|x_i) + T \ln[q_i(x_i)], \quad (4)$$

where

$$\mathbb{E}_{q_{-i}}(G|x_i) = \int dx_{-i} q_{-i}(x_{-i}) G(x_i, x_{-i}) \quad (5)$$

with $x_{-i} \triangleq [x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n]$ and $q_{-i}(x_{-i}) \triangleq \prod_{j=1, j \neq i}^n q_j(x_j)$.

⁴N.b., we do *not* project onto \mathcal{Q} but rather add a vector to get back to it. See Wolpert and Biegniewski (2004b).

For finite move spaces, the $q_i(x_i)$ component of $\eta(q)$ is independent of x_i , and given by

$$[\eta(q)]_i \triangleq \frac{1}{|X_i|} \int dx'_i [\nabla_q \mathcal{L}(q)]_{q_i(x'_i)}, \quad (6)$$

where $|X_i|$ is the number of possible moves x_i , i.e., this choice ensures that $\int dx_i q_i(x_i) = 1$ after the gradient update to the values $q_i(x_i)$.

2.3. Monte Carlo-based gradient descent and shrink-wrapping

Equation (3) gives the (negative of the) change that each agent should make to its distribution to have them jointly implement a step in steepest descent of the Maxent Lagrangian. These updates are completely distributed, in the sense that each agent's update at time t is independent of any other agents' update at that time. Typically at any t each agent i knows $q_i(t)$ exactly, and therefore knows $\ln[q_i(j)]$. However often it will not know G and/or the q_{-i} . In such cases it will not be able to evaluate the $\mathbb{E}(G|x_i = j)$ terms in Eq. (3) in closed form.

One way to circumvent this problem is to have those expectation values be simultaneously estimated by all agents by repeated Monte Carlo sampling of q to produce a set of $(x, G(x))$ pairs. Those pairs can then be used by each agent i to estimate the values $\mathbb{E}(G|x_i = j)$, and therefore how it should update its distribution. In the simplest version of this scheme such an update to q only occurs once every L time-steps. Note that only one set of Monte Carlo samples is needed for all players to determine how to update their mixed strategy, no matter how many players there are.

In this simple Monte Carlo scheme only the samples $(x, G(x))$ formed within a block of L successive time-steps are used at the end of that block by the agents to update their distributions (according to Eq. (3)). More sophisticated approaches modify the G values returned by the Monte Carlo on a player-by-player basis, etc. (Wolpert, 2003; Wolpert and Biegniewski, 2004a).

In addition, in general it may be that some agent i has no sample for the value x_i . Some approaches to address this are presented below. These approaches are fairly elaborate, being explicitly designed to be able to work for uncountable X_i . They are based on “just in time” evaluation of how to update the probability of a move, together with interpolation to estimate quantities like $\mathbb{E}(G|x_i)$.

There are a set of simpler approaches, similar to Levenberg-Marquardt descent, that are particularly appropriate for finite X_i . These approaches are known as *shrink-wrapping*. We present them here in a slightly more general context than that of minimizing the free energy.

2.4. Brouwer updating

It is possible to write down, explicitly, the q that forms the fixed point of gradient descent of the free energy. Setting $\hat{\nabla}_q \mathcal{L}(q)$ to zero in Eq. (4) gives the solution

$$q_i(x_i) \propto \exp[-\mathbb{E}_{q_{-i}}(G|x_i)/T]. \quad (7)$$

This is a set of coupled nonlinear equations for the vectors $\{q_i\}$. Brouwer's fixed point theorem guarantees the solution of Eq. (7) exists for any G (Wolpert, 2003, 2004b). Hence we call update rules based on this equation *Brouwer updating*. Note that just like gradient-descent, Brouwer updating also involves the expression $\mathbb{E}(G|x_i)$. Accordingly, when that term cannot be evaluated in closed form, it can be estimated using Monte Carlo techniques, just like in gradient-descent.

In *serial* Brouwer updating, one agent at a time updates its distribution, jumping to the optimal distribution given in Eq. (7). The order in which the agents update their distributions can be pre-fixed or random. The order can also be dynamically determined in a greedy manner, by choosing which agent to update based on what associated drop in the Maxent Lagrangian would ensue (Wolpert, 2003, 2004a; Bieniawski and Wolpert, 2004a). (These various ordering schemes are similar to the those used in the majorization and block relaxation techniques in optimization statistics.) Aside from potential effects from Monte Carlo estimation error, each step of serial Brouwer updating is guaranteed not to increase the associated (Maxent) Lagrangian.

In *parallel* Brouwer updating, this updating procedure is followed simultaneously by all agents. Accordingly, when Monte Carlo sampling is used, parallel Brouwer updating can be viewed as a type of learning in games (see Shamma and Arslan, 2004; Fudenberg and Kreps, 1993; Fudenberg and Levine, 1993). Now in general, when any q_j changes, for every $i \neq j$, what distribution q_i minimizes *is* Lagrangian will change, in accord with Eq. (7). This suggests that a step of parallel Brouwer updating may “thrash,” and have each agent change in a way that confounds the other agents’ changes. (See Shamma and Arslan (2004), Fudenberg and Levine (1993) and references therein for analysis of related issues in fictitious play.) In such a case the update may not actually decrease the associated (Maxent) Lagrangian, unlike with serial Brouwer.

There are many possible ways of addressing this problem. One is to mix parallel and serial Brouwer updating, so that only subsets of the agents perform parallel updates at any given time. These can be viewed as management hierarchies, akin to those in human organizations. Often such hierarchies can also be determined dynamically, during the updating process. Other ways to address the potential thrashing problem with parallel Brouwer have each agent i not use the current value $\mathbb{E}_{q_{-i}^t}(G|x_i)$ alone to update $q_i^t(x_i)$, but rather use a weighted average of all values $\mathbb{E}_{q_{-i}^{t'}}(G|x_i)$ for $t' \leq t$, with the weights shrinking further into the past one goes. This introduces an inertia effect which helps to stabilize the updating. (Indeed, in the continuum-time limit, this weighting becomes the replicator dynamics (Wolpert, 2004a).)

A similar idea is to have agent i use the current $\mathbb{E}_{q_{-i}^t}(G|x_i)$ alone, but have it only move part of the way the parallel Brouwer update recommends. Whether one moves all the way or only part-way, what agent i is interested in is what distribution will be optimal for i *for the next distributions of the other agents*. Accordingly, it makes sense to have agent i predict, using standard time-series tools, what those future distributions will be. This amounts to predicting what the next vector of values of $\mathbb{E}_{q_{-i}^t}(G|x_i)$ will be, based on seeing how that vector has evolved in the recent past. See Shamma and Arslan (2004) for related ideas.

2.5. Nearest Newton

Care must be taken when using second order methods to descend the Maxent Lagrangian. One approach starts by making a quadratic approximation (over the space of all distributions p , not just all product distributions q) to the Lagrangian, $\mathcal{L}(p)$ based on the current point p^t . Newton's method then specifies a p^{t+1} that minimizes that quadratic approximation. We can then find the product distribution that is nearest (in pq KL distance) to p^{t+1} and move to that product distribution. The resultant update rule for the Maxent Lagrangian is called *Nearest Newton* descent (Wolpert and Bieniawski, 2004b):

$$\begin{aligned} \frac{q_i^{t+1}(j)}{q_i^t(j)} &= 1 - S(q_i^t) - \ln(q_i^t(j)) \\ &\quad - \beta[\mathbb{E}_{q^t}(G|x_i = j) - \mathbb{E}_{q^t}(G)], \end{aligned} \quad (8)$$

where q^t is the current (assumed to be product) distribution. The conditional expectations in Nearest Newton are the same as those in gradient descent. Accordingly, they too can be estimated via Monte Carlo sampling, if need be.

In Wolpert (2004a) it is shown that in the continuum time limit, Nearest Newton updating for modifying the probability distribution of any particular agent becomes a variant of replicator dynamics (with the different strategies of replicator dynamics identified with the different possible moves of the agent performing the Nearest Newton update). That paper also shows that when the terms in that continuum limit version of Nearest Newton are Monte Carlo estimated, Nearest Newton becomes a version of fictitious play. More precisely, it becomes identical to a “data-aged” continuum time limit of parallel Brouwer updating. The stepsize of the Nearest Newton procedure is identical to the constant in the exponent of the data aging.

2.6. Private utilities

Performing the updates given above involves a separate conditional expected utility for each agent. Since accurate estimates usually require extensive sampling, the world utility G occurring in each agent's update rule can be replaced with a private utility g_i chosen to ensure that the Monte Carlo estimation of $\mathbb{E}(g_i|x_i)$ has both low bias, with respect to estimating $\mathbb{E}(G|x_i)$, and low variance (Duda et al., 2000). Bias represents the alignment between the private utility and world utility. With zero bias, updates that reduce the private utility are guaranteed also to reduce the world utility. It is also desirable for an agent to distinguish its contribution from that of the other agents: variance measures this sensitivity. With low variance, the agents can perform the individual optimizations accurately without a large number of Monte Carlo samples. The earlier COIN research explored the general concept of private utilities (Wolpert et al., 2000), providing several useful types. PC theory formalizes many of these concepts and provides a framework for developing problem specific utilities.

Two private utilities are typically used with the solution method: *Team Game* (TG) and *Wonderful Life Utility* (WLU). These utilities are defined as:

$$\begin{aligned} g_{TG_i}(x_i, x_{(i)}) &= G(x_i, x_{(i)}), \\ g_{WLU_i}(x_i, x_{(i)}) &= G(x_i, x_{(i)}) - G(CL_i, x_{(i)}). \end{aligned}$$

For TG, the private utility is simply the world utility. For WLU, the private utility is the world utility minus the world utility with the agent action “clamped” to the value CL_i . The choice of clamping value can strongly affect the performance (Wolpert et al., 2000), although clamping to the lowest probability action has been shown to be minimum variance (Wolpert, 2004b). Both of these utilities have zero bias. However, due to the subtracted term, WLU has much lower variance than TG. Other private utilities have also been explored. In particular, the *Aristocrat Utility* (AU) has been shown to be superior to WLU (Wolpert and Lee, 2004), but is often difficult to evaluate.

Often however, there is some known structure to the objective or constraint functions that can be exploited with a private utility. One example is one of the canonical problems in computer science, the k-Sat satisfiability problem (Macready and Wolpert, 2004). Other examples are problems with objective functions that are the sum of multiple contributions. A problem with this type of structure is considered later in this work.

3. Delayed sampling experiments

3.1. Basic solution algorithm

The basic optimization algorithm consists of three major components, all of which can be performed in a distributed manner: Monte Carlo sampling to obtain agent actions given their probability distributions, private utility evaluation for the agents for each joint sample, and update of the probability distributions. The latter step includes the regression to obtain the expected utility for each agent across its actions.

More precisely, the general algorithm proceeds as follows (detailed description of some of these steps are presented in the following subsections):

1. Initialize.
 - (a) Set the parameters: temperature T , probability update step size α , and data aging rate γ . For constrained problems set the Lagrange multiplier step size η .
 - (b) Assign the annealing schedule for T as a function of iteration number.
 - (c) Specify the number of Monte Carlo samples m for each iteration.
 - (d) Assign agents to the variables in the problem, with their actions representing choices for values. Set the starting probabilities for each agent to uniform over its possible actions.
 - (e) Set the termination criteria. A number of possibilities exist, including: the number of iterations or convergence of the probabilities, objective values, or actions.
2. Minimize the Lagrangian.
 - (a) Monte Carlo sample the agent probabilities to obtain m IID (identically and independently distributed) joint actions.
 - (b) For each sample,
 - Evaluate the objective function.
 - For each agent, compute the private utility.

- (c) Estimate the move-conditioned expected utilities for the agents for each of their possible moves using a regression. (An illustration is given below.)
- (d) Update the probability distributions of each agent using these estimates according to Brouwer updating or Nearest Newton. (We ensure non-negative probabilities by setting all values that are smaller than 10^{-6} to 10^{-6} , and then re-normalize to get a valid distribution.)
- (f) Update the parameters of the product distributions. Implement the assigned annealing schedule for T to update T . For continuous problems, update the regression window width τ .
- (g) Evaluate the termination criteria. If not satisfied, return to step 2(a), otherwise proceed to 3.

3. Final Evaluation.

- (a) Determine the highest probability action for each agent.
- (b) Evaluate the objective function with this set of actions.

3.1.1. Modifications for continuous variables

Many of the basic elements are easily extended to the continuous domain, such as the Monte Carlo sampling, the use of private utilities, and the basic solution algorithm. Two aspects of the solution algorithm must, however, be modified. First, the integrals of the Maxent Lagrangian are approximated using a representation for the probability density across the domain of the continuous variable. Appropriate choice of the representation maintains the favorable properties of the update rules. Several possible representations have been considered (Bierniawski, 2005).

The preferred approach is to represent the magnitudes at equally spaced locations across the variable domain using a trapezoidal approximation. Second, since the sampling occurs for only a scattering of points across the range of the variables, a regression is necessary. In the current work, a simple regression based upon exponentially weighted averaging across the sampled values is used. The specific formulation is described in detail in Section 3.1.3.

3.1.2. Data aging

A useful technique in the Monte Carlo sampling is to allow previous samples to be re-used. This is accomplished by first “aging” the older data to reflect the fact that it was formed under a different q . For the discrete problems this is simple averaging with data aging controlled by the parameter γ . One can replace the empirical average for the most recent block k by

$$\begin{aligned} E(g_i|x_i = j) &= \frac{N_{ij}^{(k)}}{D_{ij}^{(k)}} \\ &= \frac{\sum_m g_i(x_i = j, x_{(i)})\delta(x_i = j) + \gamma N_{ij}^{(k-1)}}{\sum_m \delta(x_i = j) + \gamma D_{ij}^{(k-1)}}, \end{aligned}$$

where $\delta(x_i = j)$ equals 1 when $x_i = j$ and 0 otherwise and m is the number of Monte Carlo samples in the block. To accomplish this requires storage of the matrices

N_{ij} and D_{ij} , which have dimensions number of agents by number of moves. The resulting matrix of estimated expectations has these same dimensions.

Aging typically allows the number of samples per block m to be reduced, improving the performance of the minimization. For small m though, the most recent block may have no samples of some move $x_i = j$. This is only a concern if the move has not yet been sampled because D_{ij} would be undefined. One way to avoid this issue is to force the sampling of each move at the initial iteration. Care must be taken during the forced sampling to have the $x_{(i)}$ formed by sampling $q_{(i)}$. Another alternative is to average over just those k for which $G_{i,j}(k)$ exists.

3.1.3. Regressions

For the discrete case, as just discussed, the regression for each agent consists of averaging the utility values observed for each of its actions. Data aging can then be applied to re-use information from previous samples. In the case of continuous variables, however, the sampling occurs at only a scattering of points across the range of the variables. As a result, a more complex regression is necessary.

Given m sample pairs at iteration k , $\{x_j^k, g^k(x_j^k)\}$, $j = 1, \dots, m$, the regression provides estimates of $E[g|x_i]$ for the x_i that are the locations parameterizing the distribution. In the current work, a simple regression based on exponentially weighted averaging across the sampled values is used. The regression is given by:

$$E[g^k|x_i] = \frac{N_i^k}{D_i^k} = \frac{\sum_m g^k(x_j^k) \exp\left[-(x_i - x_j^k)^2 / (2\tau^2)\right]}{\sum_m \exp\left[-(x_i - x_j^k)^2 / (2\tau^2)\right]}. \quad (9)$$

This formulation still allows data aging to be easily incorporated,

$$E[g^k|x_i] = \frac{N_i^k + \gamma N_i^{k-1}}{D_i^k + \gamma D_i^{k-1}} \quad (10)$$

with the parameter γ setting the aging rate.

One refinement to Eq. (9) is to adapt the parameter τ automatically as the optimization proceeds. Since each agent i knows the probability distribution $q_i(x_i)$ from which its x_i are drawn, it is natural to set the regression window based upon some measure associated with the distribution. For the current work, the following procedure is applied to each of the variables:

- Given the number of samples m , determine their expected locations. To do this invert the cumulative probability density function at m evenly spaced intervals. To accomplish this, use the same subroutine that generates the Monte Carlo samples.
- Determine the median spacing between adjacent expected locations.
- Set the value of τ as

$$\tau = W_0 m \overline{\Delta x}, \quad (11)$$

where $\overline{\Delta x}$ is the median spacing and W_0 is a scalar multiple, typically set to 0.2. This value proved appropriate for all the problems studied during this work. For problems involving higher numbers of variables, however, the parameter may need to be modified.

3.1.4. Annealing

The procedure for updating the temperature, referred to as the annealing schedule, plays an important role in the efficiency and reliability of the approach. If the temperature is reduced too quickly, the approach is more likely to find a local minimum. Too slowly, and a larger number of iterations, and therefore samples if Monte Carlo sampling is used, are required. In the current work, a geometric schedule is typically applied. This involves multiplying the temperature by some fixed factor every few iterations. For the various applications within this work, the specific annealing schedule is provided with each. A minimum temperature is often also employed to insure that the solutions do not become delta functions.

An advantage of casting the problem directly in terms of the Maxent Lagrangian is that the explicit annealing schedule can be avoided. For example, the saddle point of the Lagrangian can be found by performing steepest ascent of \mathcal{L} in the Lagrange parameter T while performing a descent in q . This type of approach was implemented in [Macready and Wolpert \(2004\)](#). The opportunity to include other techniques from gradient-based optimization ([Gill et al., 1981](#)) is also possible.

3.2. Example for a continuous problem

A simple two-variable continuous problem is solved to show the various aspects of the continuous domain solution method. The function is shown in Fig. 1, which consisted of several Gaussian peaks. The objective function was given by

$$\begin{aligned} G(x,y) = & 3(1-x)^2 e^{-x^2-(y+1)^2} \\ & - 10(x/5 - x^3 - y^5) e^{-x^2-y^2} \\ & - 1/3 e^{-(x+1)^2-y^2}. \end{aligned} \quad (12)$$

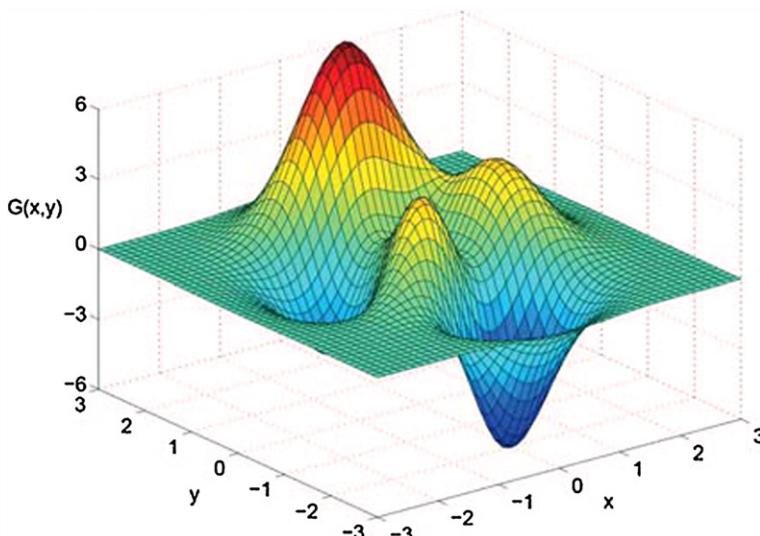


Fig. 1. Peaks function.

Table 1
Parameter settings for *Peaks* optimization

Samples per iteration	10
Initial temperature, T_o	1
Final temperature, T_f	$5 \times 10^{-2} \times E[G]$
Annealing rate	0.25 every 10 iterations
Step size, α	0.1
Data aging rate, γ	0.5
Variable ranges	$[-3, 3]$

This is also known as the *Peaks* function in MATLAB. Note that, for uniform initial probability distributions, variable x cannot see the global minimum until variable y changes its distribution away from uniform. As a result, cooperation was required between the variables to find the optimum.

Nearest Newton was used from this problem, with the parameter settings as listed in Table 1. While these values resulted in consistent convergence, they may not be the most efficient. For example, other values may result in fewer total function calls. The probability densities were represented using the trapezoidal approach with 200 points.

The overall convergence is consistent, as indicated in Fig. 2. The two plots show the convergence of the sample average and the objective value at the maximum

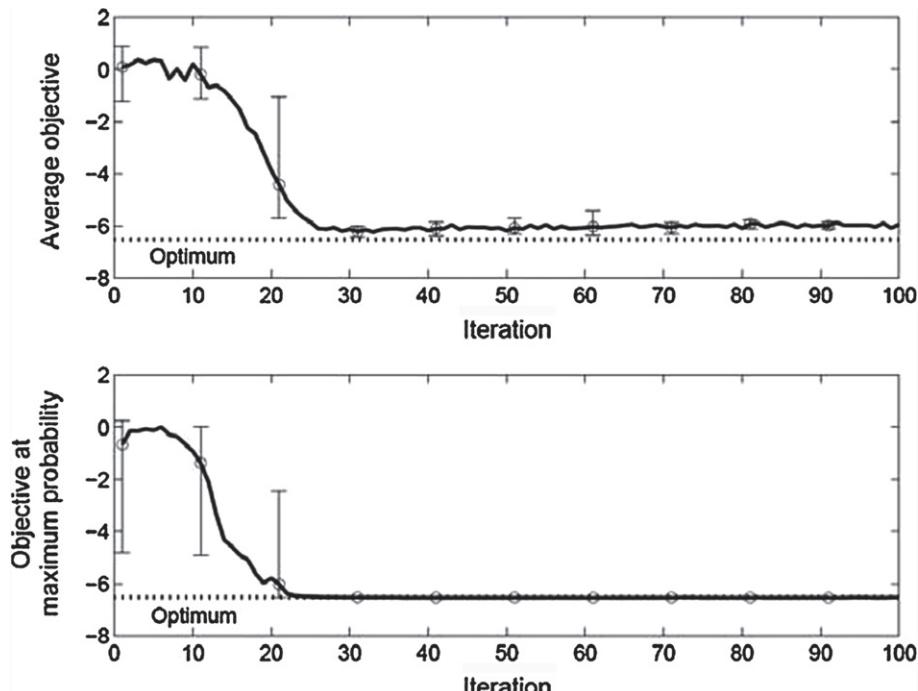


Fig. 2. Convergence history of the objective for the *Peaks* function.

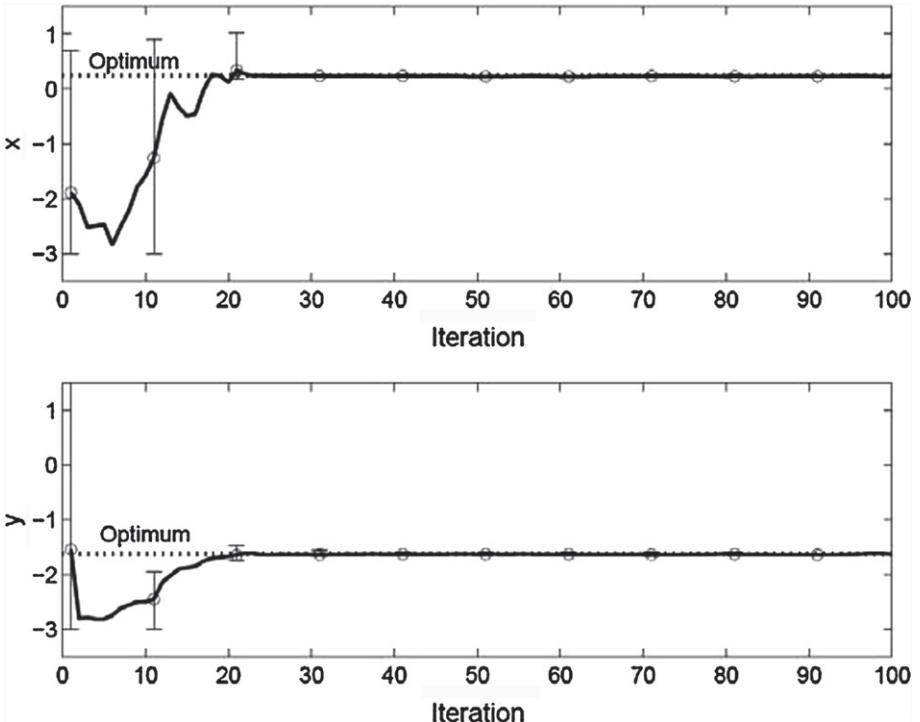


Fig. 3. Convergence history of the variables for the *Peaks* function.

probability. The histories over 10 runs were averaged with error bars showing the minimum and maximum deviations. The convergence of the variables is shown in Fig. 3. The results indicated that the technique quickly and consistently found the minimum, typically within 30 iterations. Although a small test problem, the quick convergence here with only 10 Monte Carlo samples per iteration illustrated the potential of the approach.

The converged sample average was typically slightly above the optimum in Fig. 2 because of the moderate lower bound on the temperature. A nonzero final temperature resulted in probability distributions that were centered about the optimum with some probability mass at less optimum values. As a result, when sampled, sometimes values slightly away from the optimum were obtained, reducing the overall average relative to the optimum. This is clear from Fig. 4, which shows the converged probability distributions. This figure also illustrates another advantage of the approach, the sensitivity of the objective to each of the variables. Since the converged probabilities are related to the expected utilities through Eq. (7), given the probabilities $q_x(x)$, the expected utility, $\mathbb{E}(g_x|x)$ is obtained.

The performance of the approach was dependent upon the updating of the temperature T and the regression window widths τ_x and τ_y . The corresponding parameter histories for the *Peaks* optimizations are shown in Fig. 5. While the temperature primarily followed its assigned schedule, the regression widths were

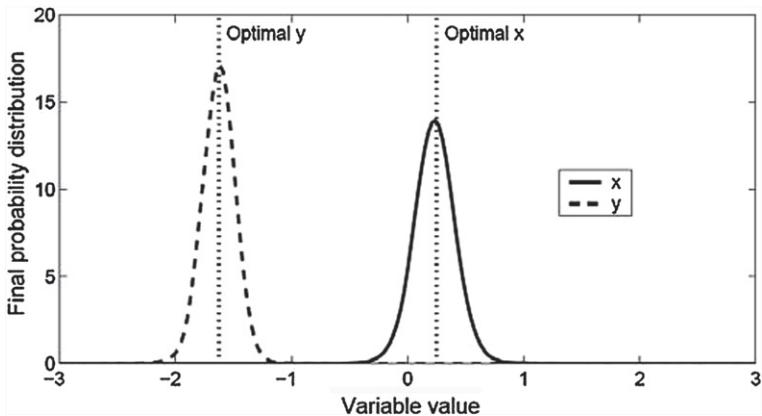


Fig. 4. Final probability distributions for *Peaks* function.

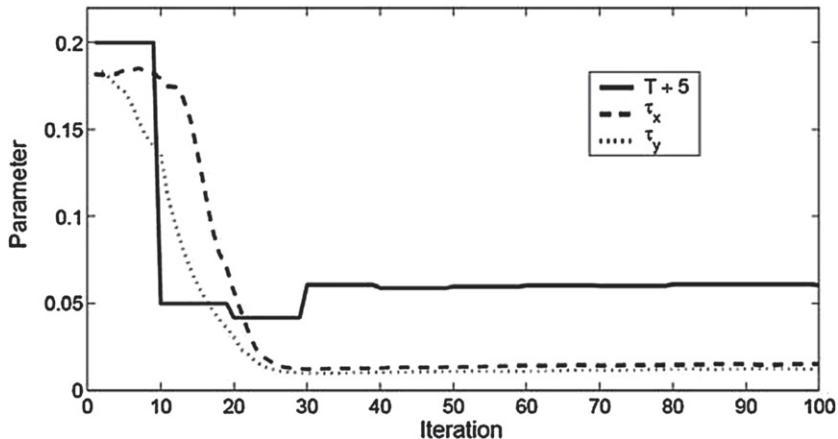


Fig. 5. Parameter history for the *Peaks* function.

updated automatically. As a result, smooth updates were obtained that closely reflected the convergence of the optimization.

3.3. Summary of performance on larger-scale problems

The approach has also been implemented on a number of larger-scale problems. In Bieniawski and Wolpert (2004a), the Probability Collectives approach using Nearest Newton was applied to three different problem domains: discrete (El Farol Bar Problem), discrete constrained (bin-packing), and constraint satisfaction (N-Queens). The performance was compared to a distributed reinforcement learning algorithm identical to Parallel Brouwer updating. This approach was at the core of the earlier COIN work, and the comparisons illustrated the improved performance, continued benefits of private utilities, and the extensions to constrained problems

obtained with Probability Collectives. [Antoine et al. \(2004\)](#) applied the approach to aircraft routing, a constrained integer programming problem. The approach solved instances involving 129 variables and 184 constraints. [Bieniawski and Wolpert \(2004b\)](#) compared the performance of the different updating schemes, Serial Brouwer, Parallel Brouwer, and Nearest Newton on many instances of the bin packing problem. Solutions within 1 bin of optimal were found for perfect packing problems involving 60 agents and 60 possible moves. Larger variants were also examined. Several variants of Serial Brouwer were investigated, including ones that attempted to reduce the cost of private utility evaluation. Nearest Newton proved to have the best performance of the approaches considered. A subset of these results are repeated later in this section. [Bieniawski et al. \(2004\)](#) investigated several aerospace related optimization problems, including a discrete structural sizing problem and a trajectory optimization. The latter results are repeated later in this section. The former addressed the 10-bar truss, a canonical discrete structural sizing problem. [Macready and Wolpert \(2004\)](#) applied the solution approach to the canonical constraint satisfaction problem, k-Sat. Private utilities exploited the sparsity structure of the problem and enabled a completely distributed, analytical solution. Hard variants with 100 variables and 430 constraints were solved.

3.4. Discrete optimization: bin packing problem

The proposed approach was applied to several variants of the bin packing problem in order to explore its application to discrete constrained optimization problems. The bin packing problem consists of assigning N items of differing sizes into the smallest number of bins each with capacity c . For the current study instances were chosen which have a designed minimum number of bins ([Falkenauer, 1994](#)) and were obtained from the OR-Library ([Beasley, 1990](#)). The instances consisted of 60 items to be packed in groups of three into 20 bins each of capacity 100. Since in general the minimum number of bins is not known, the move space of the agents was set to the number of items. The objective function for the optimization was selected as:

$$G = \begin{cases} \sum_{i=1}^N |x_i| & \text{if } x_i \leq c, \\ \sum_{i=1}^N |x_i - c| & \text{if } x_i > c, \end{cases} \quad (13)$$

where x_i is the total size of the items in bin i . Two approaches were explored to enforce the constraint on the number of bins: (i) adding a quadratic penalty to the objective or (ii) using the constraint augmented approach. For the latter constraints were added on the bin levels, $x_i \leq c$ for all i .

Figure 6 shows the comparison between all three results. The first plot shows the average number of bins over the optimum, the second number of bins over capacity, and the third plot the average objective function. Note that the unique optimum is no bins over the optimum and no bins over capacity. While Brouwer results in fewer bins it also results in more overfilled bins. In addition, the usage of the overfilled bins is much poorer than the Newton-based schemes, as indicated by the average objective (lower is better). Both Newton schemes obtain similar average objectives, with the constrained version enforcing the capacity constraints by using extra bins.

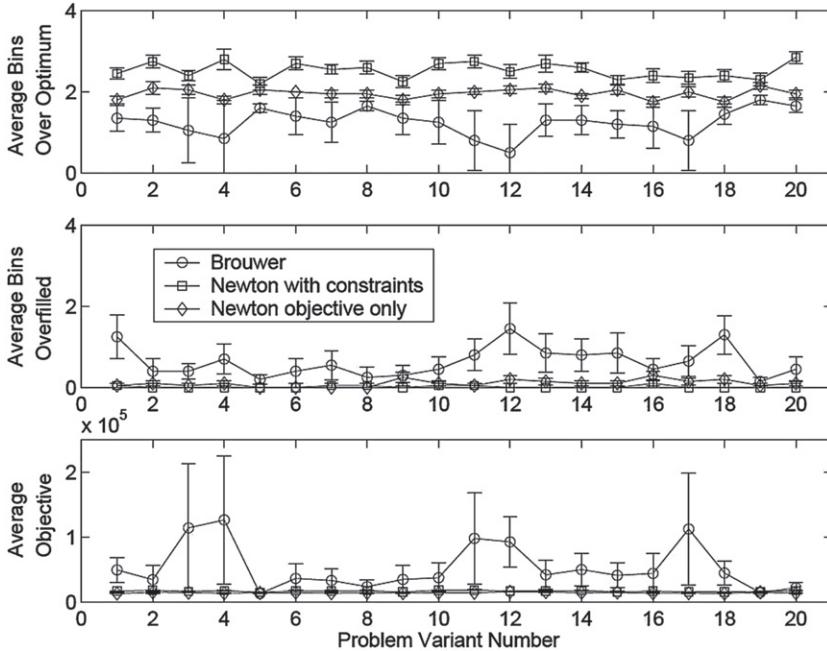


Fig. 6. Comparing RL and PD theory methods for the bin packing problem.

3.5. Collectives for control: application to stochastic collocation

Collocation is an optimization approach for systems that include a time-domain simulation, such as control synthesis. In contrast to integrating the equations of motion at each iteration for a set of the design variables, in collocation the optimization iteratively solves the simulation along with the design variables. The result is a much larger optimization problem, as the numerical integration in the simulation is replaced by constraints in the optimization. The advantage is a smooth, well-posed, and, most importantly, sparse optimization, even for problems that involve significant nonlinearities. This approach has typically found application in spacecraft and rocket trajectory problems (Hargraves and Paris, 1987), but it has also been applied to coupled design of an aircraft and its control laws (Holden, 1999).

For example, control design can be formulated as the following optimization problem:

$$\begin{aligned}
 & \text{minimize :} && h(x) \\
 & \text{with respect to :} && x \text{ and } K, \\
 & \text{subject to :} && \dot{x} = f(x, u), \\
 & \text{where :} && u = g(x, K).
 \end{aligned}$$

The objective function h , control policy g , and system dynamics f can be arbitrary. Numerous techniques are used to approximate the system dynamics to obtain constraint equations in a form compatible with an optimizer (Holden, 1999). The key

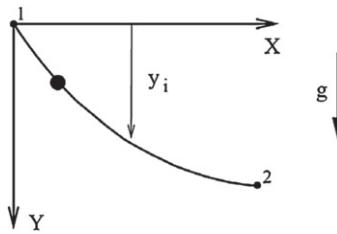


Fig. 7. The Brachistochrone problem.

advantage is that, although the problem size is now much larger since it includes the system states x , it is sparse. The collectives approach can solve this optimization efficiently by exploiting the sparse nature of the problem using a private utility. There is also the potential for collectives to extend collocation methods to account for noise and disturbances.

To illustrate the collectives approach applied to collocation, a classical calculus of variations problem, the Brachistochrone (Weisstein; Ashley, 1974), is solved. The objective is to find the minimum-time trajectory between two points for an object moving only under the influence of gravity, Fig. 7. Following (Weisstein) the objective function is:

$$G = t_{12} = \int_{x_1, y_1}^{x_2, y_2} f \, dx,$$

where

$$f = (1 + (dy/dx)^2)^{1/2} (2gy)^{1/2}.$$

Although this problem can be solved analytically, it provides a useful benchmark for evaluating optimization methods. To apply optimization techniques, variables are defined as the vertical location of points along the trajectory. The integral and the derivatives in the objective function are then approximated using these locations. A trapezoidal approximation is made to the integral at N points and a central finite difference is used for the derivative. This results in the following optimization problem:

$$\min_{\bar{y}} G = \frac{\Delta x}{2} [f_0 + 2f_1 + \cdots + 2f_N - 1 + 2f_N],$$

where for the interior points,

$$f_i = \left(1 + \left[\frac{1}{2\Delta x} (y_{i+1} - y_{i-1}) \right]^2 \right)^{1/2} (2gy_i)^{1/2}. \quad (14)$$

For the boundary points, f_0 and f_N , forward or backward approximations are used for the derivatives.

This optimization was solved by a commercially available gradient-based optimizer (The MathWorks, 2000) and by the collectives approach. Collectives are particularly applicable to objectives such as Eq. (14) due to sparse interactions between the agents representing the variables. The objective was a sum of contributions, a structure for which a private utility can be developed. Since the

individual contributions to the objective f_i were only functions of a single variable and its neighbors, a suitable private utility was

$$g_i(y_{i-2}, y_{i-1}, y_i, y_{i+1}, y_{i+2}) = \frac{\Delta x}{2} [2f_{i-1}(y_{i-2}, y_{i-1}, y_i) + 2f_i(y_{i-1}, y_i, y_{i+1}) + 2f_{i+1}(y_i, y_{i+1}, y_{i+2})]. \quad (15)$$

Equation (15) was used for the interior agents, while similar private utilities were obtained for the first and last agents. Note that this private utility has no bias since it includes all the dependencies of the objective upon agent i . This type of private utility scales well with increasing numbers of agents since the number of dependencies for each agent remains constant. Since the dependencies were known, only the other agents' probabilities, indicated in Eq. (15), were required to evaluate the expectations. This suggested two possible variants of the collectives approach. The only difference between the variants is the calculation of the expected utilities. First, sampling could be used as before, but with the agents passing their moves directly to one another so they can be evaluated according to Eq. (15). This is also equivalent to having all the agents pass their moves to a centralized entity that computes the various contributions to the objective f_i . Each agent then knows that it contributes to its own f_i and to its neighbors, f_{i-1} and f_{i+1} , using the sum of these three as its private utility. For the agents located on the edges, appropriate modifications are made. Second, the agents could pass their probability distributions to one another, which can then be integrated along with Eq. (15) to obtain the expected private utilities. This variant is now deterministic, since no sampling is involved. Results for the second approach are shown here. See Bieniawski (2005) and Bieniawski et al. (2004) for the results for the first approach.

If the agents share their probability distributions each agent now computes its conditional expected utility exactly. Figure 8 compares the iteration history for

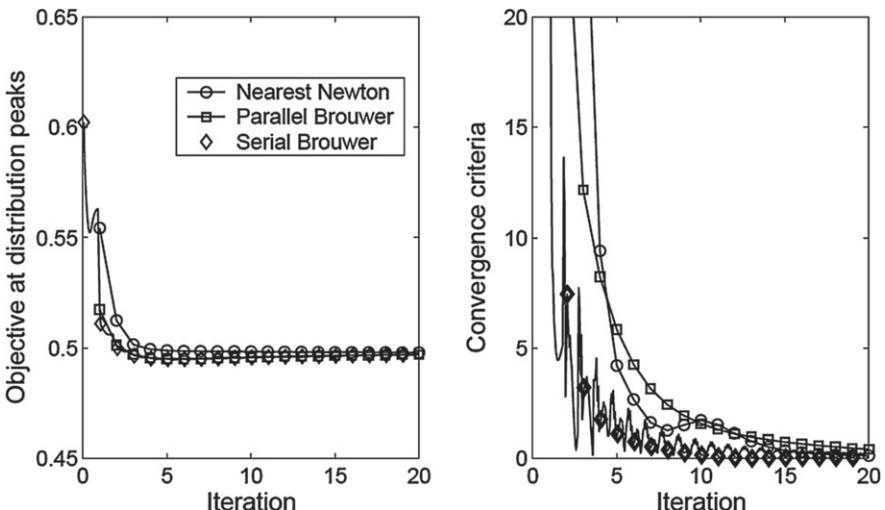


Fig. 8. Convergence history for Brachistochrone without sampling.

the different solution approaches with this implementation. The left plot shows the objective function, while the right shows the convergence criteria. The criteria used here is the 2-norm of the change in the probability distributions between iterations. The convergence of the various approaches looks very similar, and is very rapid. Serial Brouwer converged the fastest, with Parallel Brouwer finding a better optimum slightly faster than Nearest Newton. Serial Brouwer performed better since the agents performed their updates in sequence, rather than all at once as with Parallel Brouwer or Nearest Newton. As a result, the agent being updated received the most recent information from its just updated upstream neighbor. This can be implemented at no extra cost since, with no sampling, they were computationally equivalent.

The optimal trajectories for this variant are compared with the gradient-based and analytical solutions in Fig. 9. There is no appreciable difference between the discretized trajectories. The converged probability distributions for the agents are shown in Fig. 10. The figure shows the probability distributions across possible Y values for the agents vs. their position X . The distributions provide sensitivity information through their relationship to the objective via the Boltzmann distribution, Eq. (7). This was obtained without additional computational cost. The figure illustrates the higher importance of the agents closest to the starting and ending points. This partially explains differences seen between the optimal trajectories for the sampled variant. From Fig. 10 it is evident that the objective was less sensitive to the positions in the middle of the trajectory. As a result, the optimizer was slower to obtain the optimal values for these positions.

Another advantage of this approach, which still remains to be explored, is the ability to treat stochastic boundary conditions. Since these types of boundary conditions are typically represented as probability distributions, they can be incorporated as additional agents whose probabilities are not updated. These distributions could even be provided as needed by other agents to remove the need for sampling.

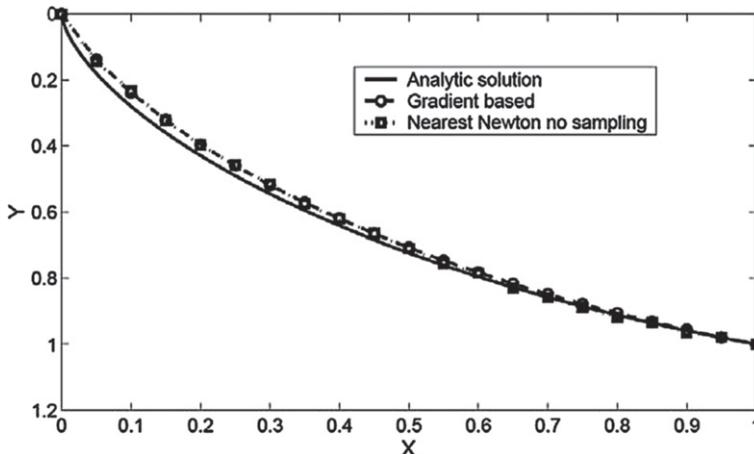


Fig. 9. Final trajectories for the Brachistochrone without sampling.

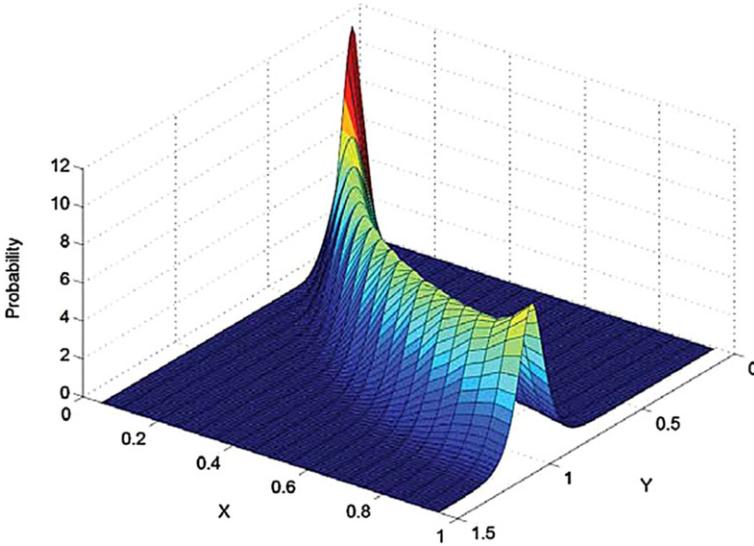


Fig. 10. Converged probability distributions for the Brachistochrone.

4. Immediate sampling theory

We now move onto immediate sampling PC. Like delayed sampling PC, it involves updating a probability distribution based on samples of that distribution. But unlike delayed sampling PC, in immediate sampling there are no restrictions on the class of probability distributions being considered.

4.1. Monte Carlo optimization

Consider the problem

$$(P1): \operatorname{argmin}_{\phi \in \Phi} \int dw U(w, \phi),$$

where even if we know the analytic form of $U(\cdot, \cdot)$, we cannot analytically perform the minimization. Monte Carlo Optimization (MCO) (Ermoliev and Norkin, 1998) is a computational way to search for the solution of (P1). In MCO we use importance sampling to rewrite the integral in (P1) as

$$\begin{aligned} \int dw U(w, \phi) &= \int dw v(w) \frac{U(w, \phi)}{v(w)} \\ &\triangleq \int dw v(w) r_{v, U, w}(\phi) \end{aligned} \tag{16}$$

for some sampling distribution v , where it is assumed that we can evaluate $v(w)$ for any w , up to an overall normalization constant. Following the usual importance sampling procedure, we then IID sample v to form a sample set $\{U(w^i, \cdot) : i = 1, \dots, N\}$. These specify a set of N sample functions from $\phi \rightarrow \mathbb{R}$:

$$r^i(\phi) \triangleq r_{v, U, w^i}(\phi).$$

In MCO, these N functions are used in combination with any prior information to estimate the solution to (P1). Conventionally, this is done by approximating the solution to (P1) with the solution to the problem

$$(P2): \quad \operatorname{argmin}_{\phi} \sum_i r^i(\phi)$$

which can often be carried out computationally. We will use the term *naive MCO* to refer to solving (P2) this way.

Note that even though $\sum_i r^i(\phi)$ is an unbiased estimate of $\int dw U(w, \phi)$ for any *fixed* ϕ , its minimizer over all ϕ is not an unbiased estimate of $\min_{\phi \in \Phi} \int dw U(w, \phi)$. This is because searching for the minimizing ϕ introduces bias; the minimizer over ϕ of $\sum_i r^i(\phi)$ is less than $\min_{\phi \in \Phi} \int dw U(w, \phi)$. Accordingly $\operatorname{argmin}_{\phi} \sum_i r^i(\phi)$ is typically a poor estimate of $\operatorname{argmin}_{\phi \in \Phi} \int dw U(w, \phi)$. This suggests that one should use some other technique than naive MCO to estimate $\operatorname{argmin}_{\phi \in \Phi} \int dw U(w, \phi)$. In particular, one might want to introduce some variance in the estimator if doing so sufficiently reduces the bias.

4.2. Naive immediate sampling

We now show how to cast the immediate sampling optimization problem as an instance of MCO. Use importance sampling to write the integral transform considered in immediate sampling, Eq. (1), as

$$\begin{aligned} \mathcal{F}_g(q_\theta) &\triangleq \int dx dg P(g|x, \mathcal{G}) F(g, q_\theta(x)) \\ &= \int dx h(x) \frac{\int dg P(g|x, \mathcal{G}) F(g, q_\theta(x))}{h(x)} \\ &= \int dx dg h(x) P(g|x, \mathcal{G}) \frac{F(g, q_\theta(x))}{h(x)} \\ &\triangleq \int dx h(x) r_{P(g|x, \mathcal{G}), h}(\theta), \end{aligned} \quad (17)$$

where we call $h(x)$ the *sampling distribution*.

We form a *sample set* of N pairs $D^1 \equiv \{x^i, g^i\}$ by IID sampling the distribution $h(x)P(g|x, \mathcal{G})$ in the integrand of Eq. (17). (That sampling is the “immediate” Monte Carlo process.) Note that D^1 is equivalent to a set of N *sample functions* from θ to \mathbb{R} :

$$r_h^i(x^i, \theta) \triangleq \frac{F(g^i, q_\theta(x^i))}{h(x^i)} : i = 1, \dots, N.$$

In the simplest version of immediate sampling, we use the functions $r_h^i(x^i, \theta)$, together with our prior knowledge (if any), to estimate the θ that minimizes $\mathcal{F}_g(q_\theta)$. As an example, not using any prior knowledge, we could estimate $\mathcal{F}_g(q_\theta)$ for any θ as

$$\int dx h(x) r_{P(g|x, \mathcal{G}), h}(\theta) \approx \frac{\sum_i r_h^i(x^i, \theta)}{N}. \quad (19)$$

For any θ , this estimate is both an unbiased estimate of $\mathcal{F}_g(q_\theta)$ and the maximum likelihood estimate of $\mathcal{F}_g(q_\theta)$. Moreover, in the usual way with importance

sampling, for a well-chosen sampling distribution this estimate will be quite accurate. Accordingly, to estimate the θ that minimizes $\mathcal{F}_{\mathcal{G}}(q_{\theta})$ we could simply search for the θ that minimizes $\sum_i r_h^i(x^i, \theta)$.⁵ However this procedure is essentially an example of naive MCO. So in particular, it would be expected to suffer from the bias problems discussed in Section 4.1.

4.3. Parametric machine learning

Parametric machine learning (PL) is a well-developed field with techniques for both supervised and unsupervised parametric learning. Broadly speaking, it seeks the solution to problems of the form

$$(P3): \operatorname{argmin}_{\xi \in \Xi} \int dx P(x) R_x(\xi)$$

for some appropriate space x , parameter set Ξ , and function R .

For example in supervised learning, $x = (x_1, x_2)$ where x_1 is the “input variable” and x_2 is the “output variable,” a “test set” is formed by sampling $P(x)$, there is a real-valued “loss function” $L(x_2, x'_2)$, and we are interested in finding the parameter ξ that minimizes

$$\int dx P(x_1) P(x_2|x_1) L(f_{\xi}(x_1), x_2) \quad (20)$$

over some parameterized set of “data fits,” $\{f_{\xi}\} : x_1 \rightarrow x_2$. This is of the form (P3) if we define $R_x(\xi) \triangleq L(f_{\xi}(x_1), x_2)$.

To perform the minimization in (P3), in PL we have a “training set” comprising N values. For example, in standard supervised learning, the training set is N pairs (x_1^i, x_2^i) formed by IID sampling $P(x)$. Although not usually viewed this way, the training set can be re-expressed as a set of N functions from $\Xi \rightarrow \mathbb{R}$, $D \equiv \{R_{x^i}(\xi)\}$.

One approach to this minimization would first make the maximum likelihood estimate

$$\begin{aligned} \int dx P(x) R_x(\xi) &\approx \frac{1}{N} \sum_i R_{x^i}(\xi) \\ &\triangleq \frac{1}{N} \sum_i R^i(\xi). \end{aligned} \quad (21)$$

One would then solve for the ξ minimizing the sum, and use this as an approximation to the solution to P3. In practice, though, this procedure is seldom used directly: although the approximation in Eq. (21) is unbiased for any fixed ξ , $\min_{\xi} \sum_i R^i(\xi)$ is not an unbiased estimate of $\min_{\xi} \int dx P(x) R_x(\xi)$. (This phenomenon is called

⁵In practice, it may make sense to divide the sum in Eq. (19) by $\sum_i 1/h(x^i)$, to reduce the variance of our estimate, and then find the θ that optimizes that ratio of sums, rather than the θ that just optimizes the numerator term (see Robert and Casella, 2004). For example, this can be helpful when one uses cross-validation to set β , as described below.

Table 2
Correspondence between variables in PL and MCO

MCO	PL
w	x
ϕ	ξ
$v(w)$	$P(x)$
$r_{v,w}(\phi)$	$R_x(\xi)$
$r_v^i(\phi)$	$R^i(\xi)$

“over-fitting.”) Therefore in PL this approximation is modified, to incorporate bias-reduction techniques.⁶

4.4. PL equals MCO

The preceding analysis suggests that the general MCO problem of extrapolation from a sample set of empirical functions to minimize the integral of Eq. (16) is identical to the general PL problem of extrapolation from a training set of empirical functions to minimize the integral in (P3). To establish this formally, make the identifications in Table 2, which result in the integrals in Eq. (16) and (P3) becoming identical.

This correspondence formally establishes that just as direct minimization of Eq. (21) is biased, naive MCO will be biased. However in PL many techniques have been found for addressing this problem of bias, and more generally for finding the optimal fit. These include in particular regularization (to reduce bias), cross-validation (to set hyperparameters of the regularization), bagging, and stacking.

The correspondence of Table 2 shows how to apply these PL techniques to MCO rather than PL, and thereby improve on naive MCO. Since immediate sampling is just a special case of MCO, we can also apply such PL techniques to improve naive immediate sampling PC. Many experiments have validated the fast optimization of such immediate sampling. In the next subsection we summarize a few of them.

5. Immediate sampling experiments

In this section, we demonstrate the application of PL to immediate-sampling PC for unconstrained optimization of both deterministic and nondeterministic functions $P(g|x, \mathcal{G})$ over real-valued x s. We first describe our choice of $\mathcal{F}_{\mathcal{G}}$, which will be pq KL distance. Next, as an illustrative example, we apply immediate sampling for this choice to an extremely simple optimization problems, where the objective $P(g|x, \mathcal{G})$ is deterministic, simply being a 2-D quadratic function. Subsequently, we apply it to

⁶Note that most sampling theory analysis of PL does not directly consider the biases and variances of the separate Monte Carlo estimators for each ξ . Rather, it considers a different type of bias and variance—the bias and variance of an entire algorithm that chooses a ξ based on associated MC estimates of expected loss. See [Wolpert \(1997\)](#) for a discussion of the relation between the two types of bias and variance.

deterministic and stochastic versions of two well-known unconstrained optimization benchmarks, the Rosenbrock function and the Woods function.

In going through these examples we highlight the use of PL techniques to enhance optimizer performance. In particular, we show that cross-validation for regularization yields a performance improvement of an order of magnitude. We then show that cross-validation for model-selection results in improved performance, especially in the early stages of the algorithm. We also show that bagging can yield significant improvements in performance.

5.1. Minimizing pq KL distance

As described in Section 2.1, a good choice of objective function for delayed sampling PC is qp KL distance where p is the Boltzmann distribution over G values, $p(x) \propto \exp(-G(x)/T)$. This qp KL distance can also be formulated as the objective \mathcal{F}_q of immediate sampling, by choosing

$$F(g, q_\theta(x)) \triangleq g q_\theta(x) + q_\theta(x) \ln[q(x)]/T$$

(cf. Eq. (1)). However in immediate sampling, rather than use the qp KL distance for this choice of Boltzmann p , it is typically better to use the pq KL distance,

$$\text{KL}(p\|q) = \int dx p(x) \ln \left(\frac{p(x)}{q(x)} \right)$$

as discussed in Rajnarayan et al. (2011). (The immediate sampling objective function can be formulated as this KL distance by choosing $F(g, q(x)) \triangleq \exp(-g/T) \ln[q(x)]$, up to an irrelevant q -independent overall multiplicative constant and an irrelevant q -independent additive constant.)

It is easy to show that when there are no restrictions on q , pq KL distance is minimized (to 0) by taking $p = q$. However, because we have a finite amount of data, we want to restrict q to some subset of all distributions. We then parametrize elements of that class by θ , and write elements of the class as q_θ . So minimizing pq KL distance is equivalent to the following cross-entropy minimization problem:

$$\begin{aligned} \text{minimize over } \theta & \quad - \int dx p(x) \ln(q_\theta(x)), \\ \text{where} & \quad \int dx q_\theta(x) = 1, \\ & \quad q_\theta(x) \geq 0 \forall x. \end{aligned} \tag{22}$$

5.1.1. Gaussian densities

If q_θ is log-concave in its parameters θ , the minimization problem (23) is a convex optimization problem. In particular, consider the case where $X = \mathbb{R}^n$, and q_θ is a multivariate Gaussian density, with mean μ and covariance Σ , parametrized as follows:

$$q_{\mu, \Sigma}(x) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp \left(-\frac{(x - \mu)^T \Sigma^{-1} (x - \mu)}{2} \right),$$

then the optimal parameters are given by matching first and second moments of p and q_θ .

$$\begin{aligned}\mu^* &= \int dx xp(x), \\ \Sigma^* &= \int dx(x - \mu^*)(x - \mu^*)^T p(x).\end{aligned}$$

It is easy to generalize this to the case where $X \subset \mathbb{R}^n$, by making a suitable modification to the definition of p . This is described in Section 5.2.1.

5.1.2. Immediate sampling with a single Gaussian

Using importance sampling, we can convert the cross-entropy integral in Eq. (23) to a sum over data points, as follows:

$$\frac{1}{N} \sum_D \frac{p(x^i)}{h(x^i)} \ln(q_\theta(x^i)),$$

where D is the data set $\{(x^i, g^i)\}, i = 1, \dots, N$. This sets up the minimization problem for immediate sampling for pq KL distance.

$$\text{minimize} - \sum_D \frac{p(x^i)}{h(x^i)} \ln(q_\theta(x^i)). \quad (23)$$

Denote the likelihood ratios by $s^i = p(x^i)/h(x^i)$. Differentiating Eq. (23) with respect to the parameters μ and Σ^{-1} and setting them to zero yields the following formulas:^{7,8}

$$\begin{aligned}\mu^* &= \frac{\sum_D s^i x^i}{\sum_D s^i}, \\ \Sigma^* &= \frac{\sum_D s^i (x^i - \mu^*)(x^i - \mu^*)^T}{\sum_D s^i}.\end{aligned}$$

5.1.3. Mixture models

The single Gaussian is a fairly restrictive class of models. Mixture models can significantly improve flexibility, but at the cost of convexity of the KL distance minimization problem. However, a plethora of techniques from supervised learning, in particular the Expectation Maximization (EM) algorithm, can be applied with minor modifications.

Suppose q_θ is a mixture of M Gaussians, that is, $\theta = (\mu, \Sigma, \phi)$ where ϕ is the mixing p.m.f., we can view the problem as one where a hidden variable z decides

⁷As one would expect, in the infinite-data limit these formulas are identical to the moment-matching results for the full-blown integral case.

⁸Note that these formulas resemble those for MAP density estimation, often used in supervised learning to find the MAP parameters of a distribution from a set of samples. The difference in this case is that each sample point is weighted by the likelihood ratio s^i , and is equivalent to “converting” samples from h to samples from p .

which mixture component each sample is drawn from. We then have the optimization problem

$$\text{minimize} - \sum_D \frac{p(x^i)}{h(x^i)} \ln (q_\theta(x^i, z^i)).$$

Following the standard EM procedure, we multiply and divide the quantity inside the logarithm by some $Q_i(z^i)$, where Q_i is a distribution over the possible values of z^i . As before, let s^i be the likelihood ratio of the i th sample.

$$\text{minimize} - \sum_D s^i \ln \left(\sum_{z^i} Q_i(z^i) \frac{q_\theta(x^i, z^i)}{Q_i(z^i)} \right).$$

Then using Jensen's inequality, we can take Q_i outside the logarithm to get a lower bound. To make this lower bound tight, choose $Q_i(z^i)$ to be the constant $p(z^i|x^i)$. Finally, differentiating with respect to μ_j , Σ_j^{-1} and ϕ_j give us the EM-like algorithm:

E-step : For each i , set $Q_i(z^i) = p(z^i|x^i)$,

$$\text{that is, } w_j^i = q_{\mu, \Sigma, \phi}(z^i = j|x^i), \quad j = 1, \dots, M.$$

M-step : Set

$$\mu_j = \frac{\sum_D w_j^i s^i}{\sum_D w_j^i},$$

$$\Sigma_j = \frac{\sum_D w_j^i s^i (x^i - \mu_j)(x^i - \mu_j)^T}{\sum_D w_j^i s^i},$$

$$\phi_j = \frac{\sum_D w_j^i s^i}{\sum_D s^i}.$$

Since this is a nonconvex problem, one typically runs the algorithm multiple times with random initializations of the parameters.

5.2. Implementation details

In this section we describe the implementation details of an iterative immediate-sampling PC algorithm that uses the Gaussian mixture models described in the previous section to minimize pq KL distance to a Boltzmann target parametrized by $\beta \equiv 1/T$. We also describe the modification of a variety of techniques from parametric learning that significantly improve performance of this algorithm. An overview of the procedure is presented in Algorithm 1.

5.2.1. Example: quadratic $G(x)$

Consider the 2-D box $X = \{x \in \mathbb{R}^2 | \|x\|_\infty < 1\}$. Consider a simple quadratic on X ,

$$G_Q(x) = x_1^2 + x_2^2 + x_1 x_2, \quad x \in X.$$

The surface and contours of this simple quadratic on X are shown in Fig. 11. Also shown are the corresponding Boltzmann target distributions p^β on X , for

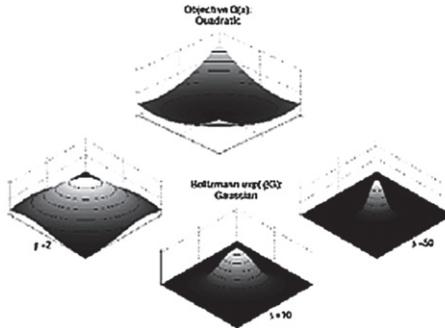


Fig. 11. Quadratic $G(x)$ and associated Gaussian targets.

Algorithm 1. Overview of pq minimization using Gaussian mixtures

- 1: Draw uniform random samples on X
 - 2: Initialize regularization parameter β
 - 3: Compute $G(x)$ values for those samples
 - 4: **repeat**
 - 5: Find a mixture distribution q_θ to minimize sampled pq KL distance
 - 6: Sample from q_θ
 - 7: Compute $G(x)$ for those samples
 - 8: Update β
 - 9: **until** Termination
 - 10: Sample final q_θ to get solution (s)
-

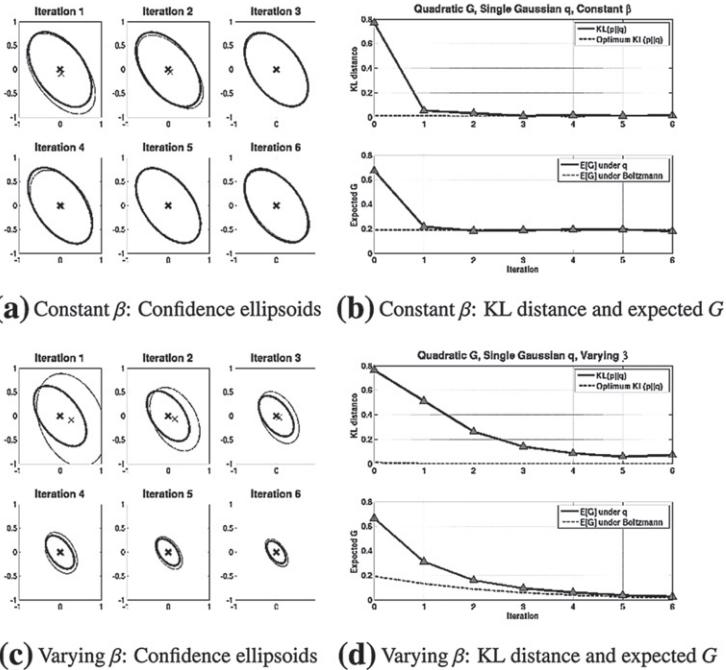
$\beta = 2, 10$, and 50 . As can be seen, as β increases, p^β places increasing probability mass near the optimum of $G(x)$, leading to progressively lower $\mathbb{E}_{p^\beta} G(x)$. Also note that since $G(x)$ is a quadratic, $p^\beta(x) \propto \exp(-\beta G(x))$ is a Gaussian, restricted to X and renormalized. We now “fit” a Gaussian density q_θ to the Boltzmann p^β by minimizing $\text{KL}(p^\beta \| q_\theta)$, for a sequence of increasing values of β . Note that q_θ is a distribution over \mathbb{R}^2 , and G_Q is not defined everywhere in \mathbb{R}^2 . Therefore, we extend the definition of G_Q to all of \mathbb{R}^2 as follows:

$$G_Q(x) = \begin{cases} x_1^2 + x_2^2 + x_1 x_2, & x \in X, \\ \infty & \text{otherwise.} \end{cases}$$

Now $p^\beta = 0$ for all $x \notin X$, and the integral for KL distance can be reduced to an integral over X . This means that samples outside X are not considered in our computations.

5.2.2. Constant β

First, we fix $\beta = 5$, and run a few iterations of the PC algorithm. To start with, we draw $N_j = 30$ samples from the uniform distribution on X . The best-fit Gaussian

Fig. 12. PC iterations for quadratic $G(x)$.

is computed using the immediate sampling procedure outlined in the preceding section. At each successive iteration, $N_j = 30$ more samples are drawn from the current q_θ and the algorithm proceeds. A total of 6 such iterations are performed. The 90% confidence ellipsoids corresponding to p^β (heavy line) and the iterates of q_θ (thin line) are shown in Fig. 12. Also shown are the corresponding values of $\mathbb{E}_{q_\theta} G(x)$, computed using the sample mean of $G_Q(x)$ for 1000 samples of x drawn from each q_θ , and $\text{KL}(p^\beta \| q_\theta)$, computed as the sample mean of $\ln(p^\beta(x)/q_\theta(x))$ for 1000 samples of x drawn according to p^β .

5.2.3. Varying β

Next, we change β between iterations, in the “update β ” step shown in Algorithm 1. With the same algorithm parameters, we start with $\beta = 10$, and at each iteration, we use a multiplicative update rule $\beta \leftarrow k_\beta \beta$, for some constant $k_\beta > 1$, in this case, 1.5. As the algorithm progresses, the increasing β causes the target density p^β to place increasing probability mass on regions with low $G(x)$, as shown in Fig. 11. Since the distributions q_θ are best-fits to p , successive iterations will generate lower $\mathbb{E}_{q_\theta} G(x)$. The 90% confidence ellipsoids and evolution of $\mathbb{E}_{q_\theta} G(x)$ and KL distance are shown in Fig. 12.

5.2.4. Cross-validation to schedule β

In more complex problems, it may be difficult to find a good value for the β update ratio k_β . However, we note that the objective $\text{KL}(p^\beta \| q_\theta)$ can be viewed

as a regularized version of the original objective, $\mathbb{E}_{q_\theta}[G(x)]$. Therefore, we use the standard PL technique of cross-validation to pick the regularization parameter β from some set $\{\beta\}$.

More precisely, at each iteration, we partition the data set D of all samples $\{(x^i, g^i)\}$ found so far, into training and test data sets, indicated by D_T and D_V , respectively. Then for each β in some candidate set $\{\beta\}$, we find the optimal parameters $\theta^\star(\beta)$ using only the training data D_T . Next, we test the associated $q_{\theta^\star(\beta)}$ on the test data D_V using the following performance measure:

$$\hat{g}(\theta) = \frac{\sum_{D_V} \frac{q_\theta(x^i)G(x^i)}{h(x^i)}}{\sum_{D_V} \frac{q_\theta(x^i)}{h(x^i)}}. \quad (24)$$

The objective $\hat{g}(\theta)$ is an estimate⁹ of the unregularized objective $\mathbb{E}_{q_\theta}[G(x)]$. Finally, we set $\beta^\star = \arg \min_{\beta \in \{\beta\}} \hat{g}(\theta^\star(\beta))$, and compute $\theta^\star(\beta^\star)$ using *all* the data D . Note that the whole cross-validation procedure is carried out without any more calls to the oracle \mathcal{G} .

We demonstrate the functioning of cross-validation on the well-known Rosenbrock problem in two dimensions, given by

$$G_R(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

over the region $X = \{x \in \mathbb{R}^2 | \|x\|_\infty < 4\}$. The optimum value of 0 is achieved at $x = (1, 1)$. The details of the cross-validation algorithm used are presented in Algorithm 2. For this experiment, we choose

$$\begin{aligned} \text{maxExtIter} &= 4, & k_1 &= 0.5, & k_2 &= 2, \\ N_j &= 10, & n_\beta &= 5, & K &= 10. \end{aligned}$$

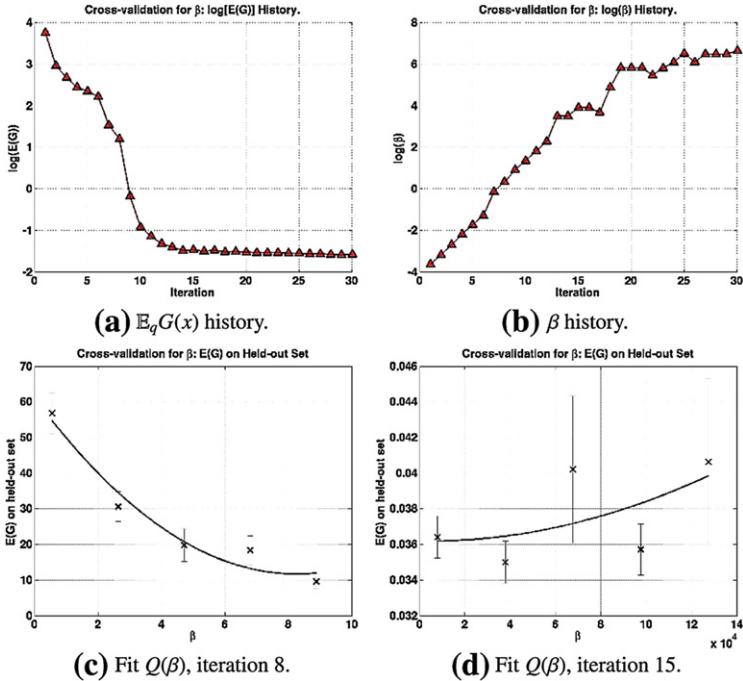
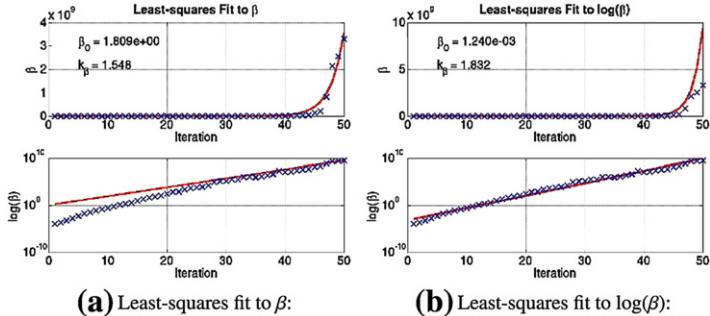
The histories of $\mathbb{E}_{q_\theta}G(x)$ and β are shown in Fig. 13. Also shown are plots of the fitted $Q(\beta)$ at iterations 8 and 15. As can be seen, the value of β sometimes *decreases* from one iteration to the next, which can never happen in any fixed multiplicative update scheme.

We now demonstrate the need for an automated regularization scheme, on another well-known test problem in \mathbb{R}^4 , the Woods problem, given by

$$\begin{aligned} G_{\text{woods}}(x) &= 100(x_2 - x_1)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 \\ &\quad + 10.1 \left[(1 - x_2)^2 + (1 - x_4)^2 \right] + 19.8(1 - x_2)(1 - x_4). \end{aligned}$$

The optimum value of 0 is achieved at $x = (1, 1, 1, 1)$. We run the PC algorithm 50 times with cross-validation for regularization. For this experiment, we used a single Gaussian q , and set.

⁹The reason for dividing by the sum of $q(x^i)/h(x^i)$ is as follows. If the training data is such that no probability mass is placed on the test data, the numerator of \hat{g}_{q_θ} is 0, regardless of the parameters of q_θ . In order to avoid this peculiar problem, we divide by the sum of $q(x^i)/h(x^i)$, as described by Robert and Casella (2004).

Fig. 13. Cross-validation for β : 2-D Rosenbrock $G(x)$.Fig. 14. Best-fit β update rule.

$$\begin{aligned} \maxExtIter &= 4, & k_1 &= 0.5, & k_2 &= 3, \\ N_j &= 20, & n_\beta &= 5, & K &= 10. \end{aligned}$$

From these results, we then attempt to find the best-fit multiplicative update rule for β , only to find that the average β variation is not at all well approximated by *any* fixed update $\beta \leftarrow k_\beta \beta$. This poor fit is shown in Fig. 14, where we show a least-squares fit to both β and $\log(\beta)$. In the fit to $\log(\beta)$ the final β is off by over 100%, and in the fit to β , the initial β is off by several orders of magnitude. We then compare the performance of cross-validation to that of PC algorithm using the fixed update rule derived from the best least-squares fit to $\log(\beta)$. From a comparison over 50 runs, we

Algorithm 2. Cross-validation for β

```

Initialize interval extension count extIter = 0, and maxExtIter and  $\beta_0$ 
repeat
    At  $\beta = \beta_0$ , consider the interval  $\Delta\beta = [k_1\beta_0, k_2\beta_0]$ 
    Choose  $\{\beta\}$  be a set of  $n_\beta$  equally-spaced points in  $\Delta\beta$ 
    Partition the data into  $K$  random disjoint subsets
    for each fold  $k$ , do
        Training data is the union of all but the  $k$ th data partitions
        Test data is the  $k$ th partition
        for  $\beta_i$  in  $\{\beta\}$ , do
            Use training data to compute optimal parameters  $\theta^\star(\beta_i, D_{T_k})$ 
            Use test data to compute held-out performance  $\hat{g}(\theta^\star(\beta_i, D_{V_k}))$ ,
            from Eq. (24)
        end for
    end for
    Compute average held-out performance,  $\bar{g}(\beta_i)$ , of  $\hat{g}(\theta^\star(\beta_i, D_{V_k}))$ 
    Fit a quadratic  $Q(\beta)$  in a least-squares sense to the data  $(\beta_i, \bar{g}(\beta_i))$ 
    if  $Q$  is convex then
        Set optimum regularization parameter  $\beta^\star = \arg \min_{\beta \in \Delta\beta} Q(\beta)$ 
    else
        Fit a line  $L(\beta)$  in a least-squares sense to the data  $(\beta_i, \bar{g}(\beta_i))$ 
        Choose  $\beta^\star = \arg \min_{\beta \in \Delta\beta} L(\beta)$ 
    end if
    Increment extIter
    Update  $\beta_0 \leftarrow \beta^\star$ 
until extIter > maxExtIter or  $Q$  is convex

```

see that using this best-fit update rule performs extremely poorly—cross-validation yields an improvement in final $\mathbb{E}_{q_\theta} G(x)$ by over an order of magnitude, as shown in Fig. 15.

5.2.5. Bagging

While regularization is a method to decrease bias, bagging is a well-known variance-reducing technique. Bagging is easily incorporated in our algorithm. Suppose, at some stage in the algorithm, we have N samples (x^i, g^i) , we resample our existing data set exactly N times with replacement. This gives us a different set of data set D' , which also contains some duplicates. We compute optimal parameters $\theta^\star(D')$. We repeat this resampling process k_b times and uniformly average the resulting optimal densities $q_{\theta^\star(D'_k)}$, $k = 1, \dots, k_b$.

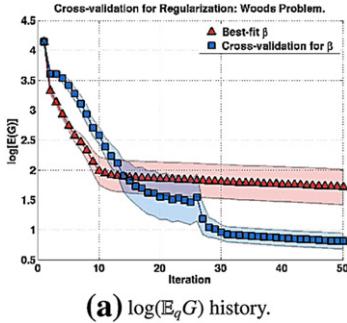


Fig. 15. Cross-validation beats best-fit fixed β update: 4-D Woods $G(x)$.

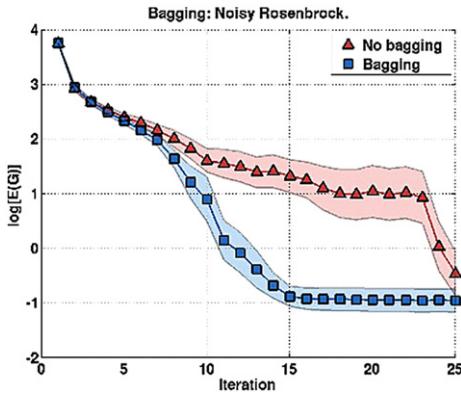


Fig. 16. Bagging improves performance: noisy 2-D Rosenbrock.

We demonstrate this procedure, using the Rosenbrock function and a single Gaussian q_θ . In this experiment, we also demonstrate the ability of PC to handle non-deterministic oracles by adding uniform random noise to every function evaluation, that is, $(g|x, G) \sim \mathcal{U}[-0.25, 0.25]$. For this experiment, $N_j = 20, k_b = 5$. The β update is performed using the same cross-validation algorithm described above. Figure 16 shows the results of 50 runs of the PC algorithm with and without bagging. We see that bagging finds better solutions, and moreover, it reduces the variance between runs. Note that the way we use bagging, we are only assured of improved variance for a single MC estimation at a given θ , and not over the whole MCO process of searching over θ .

5.2.6. Cross-validation for regularization and model selection

In many problems like the Rosenbrock, a single Gaussian is a poor fit to p^β for many values of β . In these cases, we can use a mixture of Gaussians to obtain a better fit to p^β . This raises the question of how to choose the number of components in the mixture model. To address this, we use the fact that cross-validation can be used more

broadly than just picking the regularization parameter of the training algorithm. It can be used to pick *any* set of “hyperparameters” of the training algorithm, including the hyperparameter of what model to use.

We ran a series of experiments testing this, using a greedy algorithm based on cross-validation to search over both a set of candidate β s and a set of candidate models:

1. We first pick the regularization parameter β , using Algorithm 2.
2. For that β , we use Algorithm 3 to pick the number of mixture components.

Algorithm 3. Cross-validation for model selection

Initialize the set $\{\{\phi\}\}$ of model classes $\{\phi\}$ to search over

Partition the data into K disjoint subsets

for each fold k , **do**

 Training data is all but the k th data partitions

 Test data is the k th data partition

for $\{\phi_i\}$ in $\{\{\phi\}\}$ **do**

 Compute the optimal parameter set $\theta^*(D_{T_k}) \in \{\phi_i\}$

 Compute held-out performance $\hat{g}(\theta^*(D_{V_k}))$

end for

 Compute the sample held-out performance, $\bar{g}(\{\phi_i\})$, from Eq. (24)

end for

Choose best model class $\{\phi^*\} = \arg \min_{\phi_i} \bar{g}(\{\phi_i\})$

The details of our experiments are the same as the preceding section, but without bagging. The set of models $\{\{\phi\}\}$ is the set of Gaussian mixtures with one, two, or three mixing components. Figure 17 shows the variation of $\mathbb{E}_q(G)$ vs. iteration. The mixture model is much quicker to yield lower expected G, because the Boltzmann at

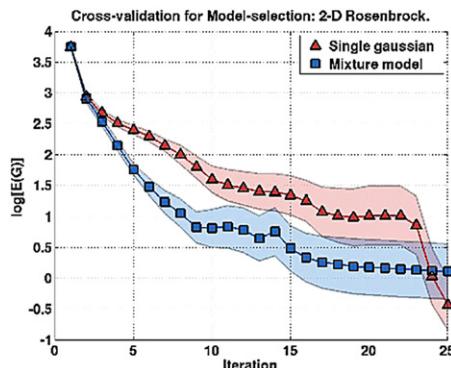


Fig. 17. Cross-validation for regularization and model-selection: 2-D penalty function $G(x)$.

many values of β is better approximated by a mixture of Gaussians. However, note that the mixture models perform poorly toward the end of the run. The reason for this is as follows: No shape regularization was used during the EM procedure. This means that the algorithm often samples from nearly degenerate Gaussians. These “strange” sample sets hurt the subsequent performance of importance sampling, and hence of the associated MCO problem. This can be alleviated by using some form of shape regularization in the EM algorithm.

5.3. Other applications of PL techniques to immediate sampling

There are numerous other PL techniques that have been successfully applied to immediate sampling. For example, cross-validation is a “winner-takes-all” technique; it chooses the single value of a “hyperparameter” like β that optimizes in-sample vs. out-of-sample behavior and then uses that value to produce a single fit to some data set. As an alternative, one can use that same in-sample vs. out-of-sample behavior to determine how best to *combine* the (fits generated using the) hyperparameters under consideration. This technique is known as *stacking* (Breiman, 1996; Clarke, 2003; Sill et al., 2009), and has been extremely successful in supervised machine learning. In Rajnarayan et al. (2011, 2007) stacking was applied for immediate sampling, and resulted in a substantial gain in performance.

Another set of techniques is related to the use of surrogate models in conventional optimization. The general idea behind these techniques is to form a single-valued fit to the data set D , $L(x)$ (e.g., using regression x is a real-valued vector), or more generally use D to form a fit $L(g|x)$ to the actual conditional distribution $P(g|x, \mathcal{G})$ (e.g., by using Gaussian processes). One could then use L to guide the setting of q_θ rather than D . The PL field of active learning in general is closely related to this, and in particular the recent work on sampling of bandits, e.g., with the UCB algorithm (Auer, 2002), is closely related.

Note though that L will not be a perfectly accurate model of the oracle in general. This implies that we should not use just L to form our next q_θ , in analogy to the work on sampling of bandits, but instead form our next q_θ after combining D and L in a way that gives more credence to D . Techniques for doing this are called *fit-based MCO/immediate sampling*. As an example, note that while samples of the actual oracle \mathcal{G} are expensive, samples of L are quite cheap. We can exploit this is to combine D and L : pick some integer k and form the union of k copies of the original training set D with a set of $k|D|$ (cheap) samples of L , and use that union rather than D as the training set for forming q_θ .¹⁰

Finally, there is ongoing work exploring what the best choice for F is. For example, arguably one is not interested in $\mathbb{E}_{q_\theta}(G) = \int dx q_\theta(x)G(x)$, but rather in *expected improvement*, which is defined as $\int_a dx q_\theta(x)G(x)$, where a is the best value g found in the current data set D . This suggests several changes to the use of qp and/or pq KL distances. Extensions of this idea are concerned not with the probability of the

¹⁰Note that if we have knowledge concerning the gradient of $G(x)$ at some x s in addition to having D s samples of $G(x)$, then we can use that extra knowledge to help set L . This is a way to incorporate gradient-based information concerning the oracle into immediate sampling.

possible g values produced by sampling $q_\theta(x)$ once, but rather with the probability of the best g value produced by sampling $q_\theta(x)$ a total of m times. (The difference is between considering mean behavior under q_θ and tail behavior under q_θ .)

Even more ambitiously, one might try to account for the fact that the next samples of q_θ will be used to update q_θ itself. After all, it is that updated version of q_θ that will subsequently be sampled, and therefore it is the samples of that updated version of q_θ that we are truly concerned with. For example, one could imagine doing this by “double-cross-validation.” (This more ambitious type of immediate sampling is also related to both the active learning and bandits literatures.) In this approach D is divided into *three* data sets, D_1, D_2 , and D_3 . One then trains on D_1 to form a q_{θ^1} using a particular value of some hyperparameter like β . Next one uses importance sampling corrections to resample D_2 to get D_2^1 , a set of unbiased samples of q_{θ^1} . After this one trains on $D_1 \cup D_2^1$ to form q_{θ^2} . Finally, one uses the empirical average of $F(g, q_{\theta^2})$ across D_3 to judge the quality of the initial hyperparameter.

6. Conclusion

This article concerns “blackbox optimization” algorithms in which one iterates the following procedure: Choose a value $x \in X$, getting statistical information about an associated value $G(x)$, then use the set of all pairs $\{(x, G(x))\}$ found so far to choose a next x value at which to sample G , the goal being to find x s with as small $G(x)$ as possible, and to do so as fast as possible. Examples of conventional blackbox optimization algorithms are genetic algorithms, simulated annealing, etc. These conventional algorithms work directly with values x , stochastically mapping the set $\{(x, G(x))\}$ to the next x . The distribution over new x s that gets sampled is never explicitly optimized. In contrast, in the Probability Collectives (PC) approach, one explicitly uses the set $\{(x, G(x))\}$ to optimize the probability distribution over x that will be sampled.

There are two general types of PC, “delayed sampling” and “immediate sampling.” In delayed sampling one restricts the distribution to be a product distribution. This allows a lot of the calculation of the optimal next distribution to be done in closed form. Delayed sampling PC is particularly well suited to distributed optimization/control problems, where a product distribution is a natural way to describe the problem.

In immediate sampling one instead reformulates the blackbox optimization problem in a way that is very similar to Monte Carlo optimization. One advantage of reformulating PC this way is that it allows us to use an arbitrary class of probability distributions, rather than being restricted to product distributions as in delayed sampling PC. Another advantage is that the reformulation can be done in a way that makes PC’s core issue, of how best to estimate a distribution based on samples, formally identical to the general problem of inductive inference considered in parametric machine learning. This formal identity allows us to apply the many very powerful techniques that have been developed in machine learning to the problem of blackbox optimization, thereby substantially improving performance.

This article reviewed some of the theory behind both kinds of PC. It then presented some of the many experiments that have demonstrated the power of both kinds of PC.

References

- Antoine, N., Bieniawski, S., Kroo, I., Wolpert, D.H., 2004. Fleet assignment using Collective Intelligence. In: Proceedings of 42nd Aerospace Sciences Meeting, AIAA-2004-0622.
- Ashley, H., 1974. Engineering Analysis of Flight Vehicles. Dover Publications, New York.
- Auer, P., Cesa-Bianchi, Fischer, P., 2002. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.* 47, 235–256.
- Bearssley, J.E., 1990. Or-library: distributing test problems by electronic mail. *J. Oper. Res. Soc.* 41 (11), 1069–1072. <<http://mscmga.ms.ic.ac.uk/info.html>>.
- Bieniawski, S., 2005. Distributed optimization and flight control using collectives. Ph.D. Thesis, Stanford University.
- Bieniawski, S., Wolpert, D.H., 2004a. Adaptive, distributed control of constrained multi-agent systems. In: Proceedings of the Third International Conference on Autonomous Agents and Multiagent Systems, pp. 1230–1231.
- Bieniawski, S., Wolpert, D.H., 2004b. Product distributions for distributed optimization. In: International Conference on Complex Systems, Boston, Mass.
- Bieniawski, S., Wolpert, D.H., Kroo, I., 2004. Discrete, continuous, and constrained optimization using collectives. In: Proceedings of 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York.
- Bieniawski, S., Kroo, I., Wolpert, D.H., 2005. Flight control with distributed effectors. In: Proceedings of 2005 AIAA Guidance, Navigation, and Control Conference, San Francisco, CA, 2005, AIAA Paper 2005–6074.
- Boyd, S., Vandenberghe, L., 2003. Convex Optimization. Cambridge University Press.
- Breiman, L., 1996. Stacked regression. *Mach. Learn.* 24.
- Clarke, B., 2003. Bayes model averaging and stacking when model approximation error cannot be ignored. *J. Mach. Learn. Res.* 683–712.
- Cover, T., Thomas, J., 1991. Elements of Information Theory. Wiley-Interscience, New York.
- De Bonet, J.S., Isbell, Jr., C.L., Viola, P., 1997. Mimic: finding optima by estimating probability densities. In: Advances in Neural Information Processing Systems, vol. 9. MIT Press.
- Duda, R.O., Hart, P.E., Stork, D.G., 2000. Pattern Classification, second ed. Wiley.
- Ermoliev, Y.M., Norkin, V.I., 1998. Monte Carlo optimization and path dependent nonstationary laws of large numbers. Technical Report IR-98-009, International Institute for Applied Systems, Analysis.
- Falkenauer, E., 1994. A hybrid grouping genetic algorithm for bin packing. In: Working Paper CRIF Industrial Management and Automation, CP 106-P4, 50 av. F.D. Roosevelt, B-1050 Brussels, Belgium.
- Fan Lee C., Wolpert, D.H., 2004. Product distribution theory and semi-coordinate transformations. In: Proceedings of the Third International Conference on Autonomous Agents and Multiagent Systems, Columbia University, pp. 522–529.
- Fudenberg, D., Kreps, D., 1993. Learning mixed equilibria. *Games Econ. Behav.* 5, 320–367.
- Fudenberg, D., Levine, D.K., 1993. Steady state learning and Nash equilibrium. *Econometrica* 61 (3), 547–573.
- Fudenberg, D., Levine, D.K., 1998. The Theory of Learning in Games. MIT Press, Cambridge, MA.
- Gill, Philip, E., Murray, Walter, Wright, Margaret, H., 1981. Practical Optimization. Academic Press.
- Hargraves, C.R., Paris, S.W., 1987. Direct trajectory optimization using nonlinear programming and collocation. *J. Guid. Control Dynamics* 10 (4).
- Holden, M., 1999. Optimization of dynamic systems using collocation methods. Ph.D. Thesis, Stanford University.
- Jaynes, E.T., 1957. Information theory and statistical mechanics. *Phys. Rev.* 106, 620.
- Koller, D., Friedman, N., 2009. Probabilistic Graphical Models. MIT Press.
- Mackay, D.J.C., 2003. Information Theory, Inference, and Learning Algorithms. Cambridge University Press.
- Macready, W., Wolpert, D.H., 2004. Distributed optimization. In: Proceedings of ICCS 04.
- Macready, W., Bieniawski, S., Wolpert, D.H., 2004. Adaptive multi-agent systems for constrained optimization. Technical Report IC-04-123, 2004.

- Meginniss, J.R., 1976. A new class of symmetric utility rules for gambles, subjective marginal probability functions, and a generalized Bayes' rule. In: Proceedings of the American Statistical Association, Business and Economics Statistics Section, pp. 471–476.
- Rajnarayan, D., Wolpert, David H., 2007. Exploiting parametric learning to improve black-box optimization. In: Jost, J. (Ed.), Proceedings of ECCS 2007.
- Rajnarayan, D., Wolpert, David H., 2008. Bias-variance techniques for monte carlo optimization: cross-validation for the ce method. <[arXiv:0810.0877v1](https://arxiv.org/abs/0810.0877v1)>.
- Rajnarayan, D., Wolpert, David H., 2011. Bias-variance trade-offs: novel applications. In: Sammut, C., Webb, G. (Eds.), Encyclopedia of Machine Learning, Springer.
- Robert, C.P., Casella, G., 2004. Monte Carlo Statistical Methods. Springer-Verlag, New York.
- Rubinstein, R., Kroese, D. 2004. The Cross-Entropy Method. Springer.
- Sabes, P., Jordan, M., 1995. Reinforcement learning by probability matching. In: Advances in Neural Information Processing Systems, vol. 8. MIT Press.
- Shamma, J.S., Arslan, G., 2004. Dynamic fictitious play, dynamic gradient play, and distributed convergence to nash equilibria. *IEEE Trans. Autom. Control* 50 (3), 312–327.
- Sill, J., Takacs, G., Mackey, L., Lin, D., 2009. Feature-weighted linear stacking. Unpublished: <[arXiv:0911.0460](https://arxiv.org/abs/0911.0460)>.
- The MathWorks, 2000. MATLAB Optimization Toolbox User's Guide.
- Weisstein, Eric W., Brachistochrone problem. From MathWorld—Wolfram Web Resource. <<http://mathworld.wolfram.com/BrachistochroneProblem.html>>.
- Wolpert, D.H., 1997. On bias plus variance. *Neural Comput.* 9, 1211–1244.
- Wolpert, D.H., 2003. Theory of collective intelligence. In: Tumer, K., Wolpert, D.H. (Eds.), *Collectives and the Design of Complex Systems*. Springer, New York.
- Wolpert, D.H., 2003. Product distribution field theory. Preprint cond-mat/0307630.
- Wolpert, D.H., 2004a. What information theory says about best response, binding contracts, and collective intelligence. In: Namatame, A. (Ed.), *Proceedings of WEHIA04*. Springer-Verlag.
- Wolpert, D.H., 2004b. Information theory—the bridge connecting bounded rational game theory and statistical physics. In: Braha, D., Minai, A., Bar-Yam, Y. (Eds.), *Complex Engineered Systems: Science Meets Technology*. Springer, pp. 262–290.
- Wolpert, D.H., Bieniawski, S., 2004a. Adaptive distributed control: beyond single-instant categorical variables. In: Skowron, A. (Ed.), *Proceedings of MSRAS04*. Springer-Verlag, pp. 1562–1567.
- Wolpert, D.H., Bieniawski, S., 2004b. Distributed control by Lagrangian steepest descent. In: *Proceedings of the IEEE Control and Decision Conference 2004*, pp. 1562–1567.
- Wolpert, D.H., Lee, C.F., 2004. Adaptive Metropolis-Hastings sampling using product distributions, cond-mat/0504163.
- Wolpert, D.H., Rajnarayan, D., 2007. Parametric learning and monte carlo optimization. <<http://lanl.arxiv.org/abs/0704.1274>>.
- Wolpert, D.H., Tumer, K., 2001. Optimal payoff functions for members of collectives. *Adv. Complex Syst.* 4 (2/3), 265–279.
- Wolpert, D.H., Tumer, K., 2002. Collective intelligence, data routing and braess' paradox. *J. Artif. Intell. Res.* 16, 359–387.
- Wolpert, D.H., Tumer, K., Frank, J., 1999. Using collective intelligence to route internet traffic. In: *Advances in Neural Information Processing Systems*, vol. 11. MIT Press, pp. 952–958.
- Wolpert, D.H., Wheeler, K., Tumer, K., 2000. Collective intelligence for control of distributed dynamical systems. *Europhys. Lett.* 49 (6), 708–714
- Wolpert, D.H., Tumer, K., Bandari, E., 2004. Improving search by using intelligent coordinates. *Phys. Rev. E* 69(017701).
- Wolpert, D.H., Strauss, C.E.M., Rajnayaran, D., 2006. Advances in distributed optimization using probability collectives. *Adv. Complex Syst.* 9.

This page is intentionally left blank

Bagging, Boosting, and Random Forests Using R

*Hansen Bannerman-Thompson¹, M. Bhaskara Rao¹,
and Subramanyam Kasala²*

¹*Department of Environmental Health, University of Cincinnati, Cincinnati,
OH 45267, USA*

²*Department of Mathematics and Statistics, University of North Carolina,
Wilmington, NC 28403, USA*

Abstract

Huge data sets are a fact of life. An impressive amount of research has been evolving with the advent of rising power of computing to extract information and signals from large noisy data. The purpose of this article is to present some machine learning tools and explain how open source software like R can be used to do your bidding. The reader needs only a rudimentary knowledge of Statistics and R. It is assumed that the reader has R installed on his computer or laptop and has some degree of familiarity with the commands of R.

Keywords: classification trees, regression trees, logistic regression, multiple regression, re-substitution error, out-of-bag error, bootstrapping, weighted bootstrapping

1. Introduction

Regression and classification are two of the most widely used statistical methodologies in science, engineering, and social research. For a data set on hand, one should be able to recognize what the response variable is and what the predictors are. In engineering parlance, the response variable is called “output” and the predictors “inputs.” Other names for the response variable are: dependent variable; effect. Alternative names for the predictors are: independent variables; causes. If the response variable or output is quantitative or numeric, regression methodologies come to the fore no matter what type the inputs are. If the response variable or output is a class label, classification techniques are the norm. There are a number

of methods available for handling such data. We have used and continue to use the following methodologies in regression in our consulting work:

1. Neural networks.
2. Traditional multiple regression.
3. Regression trees.
4. Support vector machines.
5. Bayesian networks.
6. Lasso.
7. Least angle regression.
8. Spline regression.
9. Cox regression.
10. Logic regression.

In the case of classification, the choice is equally good:

1. Neural networks.
2. Classification trees.
3. Fisher's Linear Discriminant Function.
4. Fisher's Quadratic Discriminant Function.
5. Non-parametric classification.
6. Support vector machines.
7. Logistic regression.
8. Bayesian networks.
9. Logic regression.

In the case of regression, one of the goals of building a model is to carry out prediction. The basic line of enquiry is for a given input \mathbf{x} (typically, a vector) what will be the output y ? Using training data $\{(\mathbf{x}_n, y_n); n = 1, 2, \dots, N\}$ of (input, output) on N cases, we strive to build a good model connecting the output y with input \mathbf{x} so as to reduce the prediction error as much as possible. In the case of classification, the purpose is to classify a given input into one of the set of classes. Using training data $\{(\mathbf{x}_n, y_n); n = 1, 2, \dots, N\}$ of (input, output) on N cases, we strive to put in place a good classification protocol so as to reduce misclassifications as much as possible.

Bagging, Boosting, and random forests are some of the machine learning tools designed to improve the traditional methods of model building. The purpose of the article is to present core ideas of these tools. Simple examples will be used to bring out the essence of these methods. We will demonstrate how one can use the software R to get the desired output. We present a number of data sets that are available on the Internet and other public sources for illustration.

The paper is divided into 11 sections:

- a. Introduction.
- b. Data Sets and Rationale.
- c. Bagging.
- d. Boosting.
- e. Do Bagging and Boosting really work?
- f. What is a classification tree?

- g. Classification Tree versus Logistic Regression.
- h. Random forest.
- i. Random Forest, Genetics, and Cross-Validation.
- j. Regression Trees.
- k. Boosting using the R Package, ada.
- l. Epilog.

2. Data sets and rationale

The basic goal is to find a way to connect the input \mathbf{x} with the output y . Each of the methods mentioned in the introduction provides the connection in its own way. The basic sequence of ideas in building the connection and its aftermath is illustrated using the following specific example. Later, in this section, we describe a number of data sets. All these data sets, unless otherwise specified, are available at the website <http://mlearn.ics.uci.edu/databases>, which is devoted to machine learning projects maintained by the University of California Irvine.

2.1. The heart data of San Diego Medical Center

When a heart patient is admitted, information is gathered on 19 variables x_1, x_2, \dots, x_{19} collected during the first 24 h. The variables include blood pressure, age, and 17 other ordered binary variables summarizing the medical symptoms considered as important indicators of the patient's condition. The goal of collecting data is to develop a method of identifying high-risk patients (those who will not survive at least 30 days) on the basis of the initial 24-h data. The data consist of information on 779 patients. Of these, 77 patients died within 30 days of admittance and the remaining were survivors.

This is a problem in classification. For a heart patient admitted to the hospital, let

$$\begin{aligned}y &= 1 \text{ if the patient dies within 30 days of admittance,} \\&= 0 \text{ if the patient survives at least 30 days.}\end{aligned}$$

The y -variable is a class variable. For each patient in the data set, we have an input $\mathbf{x} = (x_1, x_2, \dots, x_{19})$ and an output y . Using the data $\{(x_n, y_n); n = 1, 2, \dots, 779\}$ on 779 patients, we want to develop a classification protocol. The protocol, technically, is of the form

$$y = f(\mathbf{x}),$$

where for each input \mathbf{x} , $y = 1$ or 0 . In other words, a classification protocol is a map from the input space typified by the vector \mathbf{x} and the output space is $\{0, 1\}$ represented symbolically by y .

Once the classification protocol f is in place, we want to judge how good the protocol is. One can judge the protocol using the training data. For Patient No. 1 in the training data, we know the input \mathbf{x}_1 (all the 19 measurements) and output y_1 (whether or not he/she died within 30 days). We can find out what the classification protocol f says to the input \mathbf{x}_1 . Find out what the y value is for the input \mathbf{x}_1 . In other words, determine $\hat{y}_1 = f(\mathbf{x}_1)$. This means that we are predicting what will happen

to the patient using the classification protocol f . We do know what happened to the patient (y_1) precisely. The entities y_1 and \hat{y}_1 may not agree. If they agree, the protocol classified Patient No. 1 correctly. If not, we say that a misclassification occurred. We can execute these steps for every patient in the data set. We can calculate the total number of misclassifications and the misclassification rate. This misclassification rate is used to judge the quality of the classification protocol. Thus there are two important features in the life of a protocol: development and its quality.

There is another way to judge the quality of a given classification protocol. Typically a particular method is used in building a classification protocol. Choose randomly a certain percentage, say 80%, of the given hard data, build a classification protocol using the chosen method, feed the remaining 20% of the data into the classification protocol, and finally calculate the misclassification rate. Repeat this exercise many, many times and then average the rates over the repeats. The average misclassification rate is called a cross-validation estimate of the rate. One can provide a confidence interval for the true misclassification rate using the empirical distribution of the misclassification rates calculated over the repeats.

What is the ultimate goal for building a classification protocol? Suppose we have a new heart patient wheeled into the hospital. In the next 24 h we will gather information on 19 variables x_1, x_2, \dots, x_{19} . The doctors in charge are curious: is he/she going to survive at least 30 days? Check what the protocol says. Feed the input into the protocol. See how the patient is classified. If the classification protocol is reliable, i.e., low misclassification rate, the protocol may be helpful in identifying high-risk patients.

The classification protocol in place could be very revealing. As a byproduct, we will try to understand among the 19 inputs which ones are the most dominant or influential in the classification. We will take up this issue later when we are discussing a specific example (Iris data).

There are many methods in building classification protocols. A partial list is provided in the Introduction. We will elaborate on one of these methods of classification, namely, classification tree methodology. Jumping ahead, the tools we will be presenting are designed to improve existing classification methodologies. In a seminal paper Breiman (1996) proposed a new method of improving a classification protocol. We will be looking into this method shortly. There are other data sets too for discussion.

2.2. Breast Cancer data of University of Wisconsin Hospitals

The researchers at the hospitals collected data on 699 patients. For each patient with suspected lump in breast, a sample of breast tissue is sent to a lab for analysis. The result is either the lump is benign or malignant. Biopsy is a gold standard procedure. The goal is to avoid implementing the painful gold standard procedure for diagnosis of cancer. Are there other simple measurements that are informative of cancer? On each patient measurements on nine variables consisting of cellular characteristics are taken.

Input

- x_1 = Clump thickness on a scale (1–10).
- x_2 = Uniformity of cell size (1–10).
- x_3 = Uniformity of cell shape (1–10).

- x_4 = Marginal adhesion (1–10).
- x_5 = Single epithelial cell size (1–10).
- x_6 = Bare nuclei (1–10).
- x_7 = Bland chromatin (1–10).
- x_8 = Normal nucleoli (1–10).
- x_9 = Mitosis (1–10).

Output

- $y = \text{benign}$ if the clump is benign,
- $= \text{malignant}$ if the clump is malignant.

The output y can be coded as 0 or 1 or, for that matter 2 or 4. It is not important how the output is coded. What is important is to note that the output is a class variable, or more specifically, a binary variable. In the jargon of R, it is a factor. The measurements on the input variables are easy to procure. If we know the input $\tilde{x} = (x_1, x_2, \dots, x_9)$ on a patient, can we predict the output? This is again a classification problem. We will take up this example later to illustrate some of the data mining techniques.

2.3. Diabetes data of Pima Indians (*The National Institute of Diabetes and Digestive and Kidney Diseases*)

The data base consists of 768 cases, eight input variables, and two classes. The variables are some specific medical measurements on the patient plus age and pregnancy information. The classes are: tested positive for diabetes (268) or negative (500). The inputs are:

- x_1 = Number of times pregnant
- x_2 = Plasma glucose concentration
- x_3 = Diastolic blood pressure
- x_4 = Triceps skin fold thickness
- x_5 = 2-h serum insulin
- x_6 = Body mass index
- x_7 = Diabetes pedigree function
- x_8 = Age

2.4. Spliced DNA sequence data of the Genbank

For each subject in the sample, a certain window of 60 nucleotides is looked at.

Input

- x_1 = The nucleotide in the first position of the window (possible values: A, G, C, T)
- x_2 = The nucleotide in the second position of the window
- ...
- x_{60} = The nucleotide in the 60th position of the window

Output

Look at the middle of the window. It is classified as an intron/exon boundary, exon/intron boundary, or neither. The output is a class variable with three classes.

This is again a classification problem with the number of classes equal to three.

2.5. Boston housing

It has 506 cases corresponding to census tracts in the greater Boston area.

Input

It has 13 predictor variables.

Output

The y -variable is the median housing price in the tract. The response variable is quantitative. This is a regression problem.

Input

- x_1 = Per capita crime rate
- x_2 = Proportion of residential land zoned for lots over 25,000 sq.ft.
- x_3 = Proportion of non-retail business acres
- x_4 = 1 or 0 (1 if the tract bounds Charles River; 0 otherwise)
- x_5 = Nitric oxide concentration
- x_6 = Average number of rooms per dwelling
- x_7 = Proportion of owner-occupied units built prior to 1940
- x_8 = Weighted distances to five Boston employment centers
- x_9 = Index of accessibility to radial highways
- x_{10} = full-value property-tax rate per \$10,000
- x_{11} = Pupil-teacher ratio
- x_{12} = $1000 * (Bk - 0.63)^2$, where Bk is the proportion of blacks
- x_{13} = % lower status of the population

2.6. Ozone data

The data consist of 330 complete readings of maximum daily ozone at a hot spot in the Los Angeles basin and eight predictor variables—all meteorological, i.e., temperature, humidity, etc.

Output

y = Maximum ozone

Input

- x_1 = Wind speed
- x_2 = Humidity
- x_3 = Temperature
- x_4 = Inversion base height
- x_5 = Daggot pressure gradient
- x_6 = Inversion base temperature
- x_7 = Visibility
- x_8 = Vandenberg 500 mb height

2.7. Soybean data

The data consist of 307 cases, 35 variables (inputs), and 19 classes (output). The classes are various types of soybean diseases. The inputs are observations on the plants together with some climate variables.

There is a website devoted to Boosting www.boosting.org. The website has tutorials and research papers. A more detailed description of the data sets can be obtained from the website.

3. Bagging

Schematically, this is how bagging is initiated. Start with a data set and understand the goals behind data collection. Identify whether it is a classification problem or a regression problem. Choose and fix an appropriate regression or classification method. Implement then the bagging procedure adopting the chosen method. The rudiments of bagging are outlined below.

We have data on a response variable y and a predictor \mathbf{x} for N individual cases. The predictor \mathbf{x} is typically multi-dimensional and the response variable y is either quantitative or a class variable. The data are denoted by

$$\mathcal{L} = \{(\mathbf{x}_n, y_n); n = 1, 2, \dots, N\}.$$

The entity \mathcal{L} is called learning or training set. The primary goal in data analysis is to build a model

$$y = \varphi(\mathbf{x}, \mathcal{L})$$

connecting the predictor \mathbf{x} to the response variable y using the learning set \mathcal{L} . Once we have the model in place, we can predict the response y for any given input \mathbf{x} . The predicted value of y is $\varphi(\mathbf{x}, \mathcal{L})$.

We assume that we have a method to produce the predictor $\varphi(x, \mathcal{L})$. We might have used a neural network, traditional multiple regression methodology in the case the response variable is quantitative, logistic regression in the case the response variable is binary, a support vector machine, a regression tree, a classification tree, etc. We choose our own methodology we are comfortable with to carry out the task of developing the prediction equation $y = \varphi(x, \mathcal{L})$. The method is fixed throughout what follows.

Suppose we are given a sequence \mathcal{L}_k , $k = 1, 2, \dots, M$ of training sets. Each training set has N cases generated in the same way as the original one \mathcal{L} . These training sets are replicates of \mathcal{L} . Using our own fixed chosen methodology, we build the prediction equation

$$y = \varphi(x, \mathcal{L}_k)$$

for each training set \mathcal{L}_k , the same way we built $\varphi(x, \mathcal{L})$. Thus we will have M prediction equations. It is natural that we should combine all prediction equations into a single one.

Suppose the response variable y is quantitative. Define for any input \mathbf{x} ,

$$\varphi_A(x) = (1/M) \sum_{k=1}^M \varphi(x, \mathcal{L}_k).$$

We are averaging all the predictors! The subscript A stands for “aggregation.”

Suppose the response variable y is a class variable. Let the classes be denoted by $1, 2, \dots, J$.

For a given input \mathbf{x} , let

$$\begin{aligned}N_1 &= \#\{1 \leq k \leq M; \varphi(\mathbf{x}, \mathcal{L}_k) = 1\}, \\N_2 &= \#\{1 \leq k \leq M; \varphi(\mathbf{x}, \mathcal{L}_k) = 2\}, \\&\dots \\N_J &= \#\{1 \leq k \leq M; \varphi(\mathbf{x}, \mathcal{L}_k) = J\}.\end{aligned}$$

The entity N_1 is the number of prediction equations each of which classifies the object \mathbf{x} into Class 1, etc.

Define for an input \mathbf{x} ,

$$\varphi_A(\mathbf{x}) = i, \quad \text{if } N_i \text{ is the maximum of } \{N_1, N_2, \dots, N_J\}.$$

The object \mathbf{x} is classified into Class i if the majority of the prediction equations classify \mathbf{x} into Class i . In other words, the aggregate classification is done by voting. Whichever class gets a majority voting, that is the class the input \mathbf{x} is classified into.

Usually, we have a single learning set \mathcal{L} without the luxury of replicates of \mathcal{L} . In order to generate replicates of the given training set, we resort to bootstrapping.

3.1. Training set

Treat the training set as population.

Population	(\mathbf{x}_1, y_1)	(\mathbf{x}_2, y_2)	\dots	(\mathbf{x}_N, y_N)
Prob.:	1/N	1/N	\dots	1/N

Our population consists of N units. The i th unit is identified with the entity (\mathbf{x}_i, y_i) . Draw a random sample of size N from the population *with replacement*. This is called a bootstrap sample. Denote this bootstrap sample by $\mathcal{L}^{(1)}$. Using the replicate $\mathcal{L}^{(1)}$, build the predictor $\varphi(x, \mathcal{L}^{(1)})$ just the way we built $\varphi(x, \mathcal{L})$ using the *same methodology*.

Bootstrap again and again generating M bootstrap samples $\mathcal{L}^{(1)}, \mathcal{L}^{(2)}, \dots, \mathcal{L}^{(M)}$. Build the corresponding predictors φ s. The construction of prediction equation requires a clarification. Suppose y is a class variable. For example, suppose we use classification tree methodology to build the prediction equation $y = \varphi(x, \mathcal{L})$. We should use the classification tree methodology to build the prediction equation for each bootstrap sample $\mathcal{L}^{(k)}$. Carry out aggregation in the usual way as outlined above.

This procedure is called “bootstrap aggregating.” The acronym “bagging” is used for this procedure.

This is not a new method of classification. This is an enhancement of whatever method we used in building a prediction equation.

Evidence, both experimental and theoretical, suggests that bagging improves the accuracy of the chosen method of building prediction equations. Here accuracy means correct classification rate.

When does bagging improve accuracy? It depends how stable the selected method is. A methodology is said to be unstable if small changes in the training set \mathcal{L} result in big changes in the prediction equation $y = \varphi(x, \mathcal{L})$.

Breiman (1996a,b, 1998) showed that neural networks, regression trees, classification trees, and subset selection in the linear regression are unstable. The k -nearest neighbor method is stable.

If our chosen method is unstable, bagging improves accuracy. This is intuitively clear. If the methodology is unstable, a bootstrap sample of \mathcal{L} will induce big changes in the prediction equation $y = \varphi(x, \mathcal{L})$. Therefore, aggregation of repeated bootstrap samples can only improve the accuracy of the prediction equation. If your chosen method is stable, bagging may not improve accuracy. We now illustrate the working of the bagging procedure using the Breast Cancer data of Wisconsin Hospitals. The relevant R commands and annotation are given below:

1. Download the package “ipred” onto your computer from the main website of R. This is the package we will use for bagging. There are other packages which can be used for bagging.
2. Activate the package (`> library(ipred)`).
3. This package has a number of data sets including the Wisconsin data on Breast Cancer. For a list of data sets, type at the prompt `> data(package = "ipred")`. The list of data sets pops up as a separate page.
4. For a documentation on the package, type `> ?ipred`. The documentation outlines the basic commands of bagging and a discussion of the inputs and outputs. More importantly, the usage of the commands is illustrated by some examples. It is useful to work out the examples to get a feeling how the commands operate and what output is reported.
5. Download the Breast Cancer data (`> data(BreastCancer)`).
6. Ascertain the dimension of the data (`> dim(BreastCancer)`).
7. Ascertain how the data are presented. Is it in the format of a data.frame or matrix? (`> class(BreastCancer)`). Typically, the columns are variables and rows are individuals.
8. Look at the top six rows of the data (`> head(BreastCancer)`). It is always a good idea to get to know your data.
9. What is the nature of each variable? Is it numeric, factor, or integer? (`> sapply(BreastCancer, class)`). This is a useful step when we want to predict the response variable y for a new individual.
10. The basic R command for bagging is bagging. The basic inputs are: formula, data, coob, comb, nbagg. In the formula, one spells out the response variable and the covariates. If the response variable is a factor, bagging treats the problem as a classification variable. If the response variable is numeric, bagging treats the problem as a regression problem. The identification of the response variables is built into the data. If the input is `coob = TRUE`, the output gives the out-of-bag error rate (misclassification rate) in the case of categorical response variable. Otherwise, it reports the root mean square error. The input “comb” lets you choose your methodology on which bagging is to be built. If the method is not specified, the tree methodology of Breiman et al. (1984) is the default. The package “rpart” executes this methodology. This package is an integral part of the “ipred” package. There are other packages “MASS,” “mlbench,” “survival,” “splines,” “nnet,” and “class” built into “ipred.” Each package specializes in one or more methods of classification and regression. For example, linear discriminate analysis (lda) facility is available in “MASS” and neural networks facility is available in “nnet.” Finally, the input “nbagg” lets you choose the number of bootstrap samples. The default is set at 25. The R command

“bagging” is executed on the Wisconsin Breast Cancer data and output discussed. The commentary is reported in italics preceding the R command.

Download the data.

```
> data(BreastCancer)
```

The data has 699 rows and 11 columns.

```
> dim(BreastCancer)
```

```
[1] 699 11
```

Look at the top six rows of the data.

```
> head(BreastCancer)
```

	Id	Cl.thick- ness	Cell.size	Cell. shape	Marg. adhesion	Epith. c.size
1	1000025	5	1	1	1	2
2	1002945	5	4	4	5	7
3	1015425	3	1	1	1	2
4	1016277	6	8	8	1	3
5	1017023	4	1	1	3	2
6	1017122	8	10	10	8	7
	Bare. nuclei	Bl. cromatin	Normal. nucleoli	Mitoses	Class	
1	1	3	1	1	benign	
2	10	3	2	1	benign	
3	2	3	1	1	benign	
4	4	3	7	1	benign	
5	1	3	1	1	benign	
6	10	9	7	1	malignant	

The first column is the id of the patient. This is of no use in our data analysis. The last column “Class” is our response variable. All other columns are the predictors of the problem.

A folder (MBTree, the name is your choice) is created for the storage of the output. The input “data = BreastCancer[, -1]” creates a data folder out of the data “BreastCancer” with all rows intact and the first column omitted. The model statement “Class ~ .” means the response variable is “Class” and the covariates are the rest of the columns in “BreastCancer[, -1].” If you want to focus on a subset of the full list of covariates, you need to type these variables in the model statement additively by inserting the + sign between the variables. For example, if you want to entertain only the covariates Mitoses and Normal.nucleoli, the model statement will be Class ~ Mitoses + Normal.nucleoli. The input “coob = T” will give the out-of-bag error rate, where T stands for TRUE. The basic command for bagging in R is bagging.

```
> MBTree <- bagging(Class ~ ., data = BreastCancer[, -1], coob = T)
```

In the heart of the output lie 25 classification trees (default) a la Breiman, Friedman, Olshen, and Stone. Each patient’s data on the nine covariates is fed into each tree.

Each tree then classifies the subject as “benign” or “malignant.” Following the majority voting, the subject is classified as “benign” or “malignant.” The true status of the patient is known. We can assess whether or not the subject is correctly classified. The out-of-bag error rate is precisely the percentage of patients misclassified by the trees collectively. The error rate is an indication how much bagging is useful in classifying a new patient on whom we have data on the nine covariates. The output stored in the folder “MBTree” summarizes the essentials of the bagging procedure implemented on our data.

```
> print(MBTree).
```

Bagging classification trees with 25 bootstrap replications

Call: bagging.data.frame(formula = Class ~ ., data = BreastCancer[, -1], coob = T)

Out-of-bag estimate of misclassification error: 0.0439

The misclassification rate is 4.39%. We could implement the “rpart” command on our data and we can calculate the misclassification rate from the single tree thus produced. We can then assess the efficacy of a single tree for classification vis-à-vis the collective wisdom of 25 trees.

Now we enter a crucial phase of prediction. We have a new patient on whom data are available on all the nine covariates. For the sake of illustration, assume that the new patient has the level “5” on each covariate. We want to predict whether the patient’s tumor is benign or malignant. We need to feed the data into every tree sitting in the inner recesses of the computer and check on their collective wisdom in classifying the patient. We create a data folder incorporating the information on the patient. We need to know the status of each covariate in the original data folder.

```
> class(BreastCancer)
[1] "data.frame"

> sapply(BreastCancer, class)
$Id
[1] "character"
$Cl.thickness
[1] "ordered" "factor"
$Cell.size
[1] "ordered" "factor"
$Cell.shape
[1] "ordered" "factor"
$Marg.adhesion
[1] "ordered" "factor"
$Epith.c.size
[1] "ordered" "factor"
$Bare.nuclei
[1] "factor"
$Bl.cromatin
```

```
[1] "factor"
$Normal.nucleoli
[1] "factor"
$Mitoses
[1] "factor"
$Class
[1] "factor"
```

All the covariates are definitely factors (categorical). We thus create a new data folder accordingly.

```
> NewData <- data.frame(Cl.thickness = "5", Cell.size = "5", Cell.shape = "5",
Marg.adhesion = "5", Epith.c.size = "5", Bare.nuclei = "5", Bl.cromatin = "5",
Normal.nucleoli = "5", Mitoses = "5")
```

The file “NewData” has the following information.

```
> NewData
```

Cl. thickness	Cell. size	Cell. shape	Marg. adhesion	Epith. c.size	Bare. nuclei
5	5	5	5	5	5
Bl. cromatin	Normal. nucleoli	Mitoses			
5	5	5			

Predict the class of the new patient

```
> MBPredict1 <- predict(MBTree, NewData)
> MBPredict1
[1] malignant
Levels: malignant
```

The new patient is classified as having “malignant” tumor. Note that the input, for example, is Cl.thickness = “5” not Cl.thickness = 5. One can feed the data on any patients in our original data set into our trees and check on the classification vis-à-vis what we know already. We feed the data on the first six patients into the trees.

```
> MBPredict1 <- predict(MBTree, BreastCancer[1:6, ])
> MBPredict1
[1] benign benign benign benign benign malignant
Levels: benign malignant
```

There is no misclassification here.

The bagging procedure is purely exploratory. No statistical inference is possible from the output. One can inject some statistics into the bagging procedure by conducting cross-validation. One way to conduct cross-validation proceeds as follows. Select at random 80% of the sample without replacement. This is a training

sample. Apply the bagging procedure on the training sample. The remaining 20% of the sample is our test data. Calculate the misclassification error rate by feeding the test data into the trees. Repeat this procedure 5000 times, say. Thus we will have a bootstrap distribution of the error rate from which we can set up a 95% bootstrap confidence interval for the true error rate.

4. Boosting

Let us explain the “Boosting” methodology in the context of a binary classification problem. Before we invoke the “Boosting” paradigm, we must choose a classification methodology.

Let us get back to a binary classification problem. We have a training sample of size N . We need to develop a classification protocol based on the training sample.

4.1. Data (training sample)

$$(\mathbf{x}_1, y_1) \quad (\mathbf{x}_2, y_2) \quad \dots \quad (\mathbf{x}_N, y_N)$$

Each \mathbf{x}_i is an n -component vector. Each y_i is either +1 or -1. The entity -1 signifies that the i th individual belongs to Group 1 and +1 indicates that the individual belongs to Group 2.

Step 1: Write down the training sample with uniform weights.

$$\text{Sample: } (\mathbf{x}_1, y_1) \quad (\mathbf{x}_2, y_2) \quad \dots \quad (\mathbf{x}_N, y_N)$$

$$\begin{array}{llll} \text{Weights: } & w_{11} & w_{12} & \dots & w_{1N} \\ & 1/N & 1/N & \dots & 1/N \end{array}$$

Use the preferred method of classification to find an optimal classifier f_1 . Freund and Schapire (1966), the originators of “Boosting,” used Neural Networks. Statisticians usually use CART (Classification and Regression Trees). At this stage, the weights play no role in the development of the classifier. We set each weight w_{1j} equal to $1/N$ in conformity with what follows.

Calculate the error rate err_1 associated with the classifier f_1 . (We are judging how good the classifier is.)

If $f_1(\mathbf{x}_1) = y_1$ (correct classification), error is zero.

If $f_1(\mathbf{x}_1) \neq y_1$ (wrong classification), error is one.

Etc.

<i>Observation</i>	<i>Error</i>	<i>Weight</i>
(\mathbf{x}_1, y_1)	e_{11} (1 or 0)	w_{11}
(\mathbf{x}_2, y_2)	e_{12} (1 or 0)	w_{12}
...
(\mathbf{x}_N, y_N)	e_{1N} (1 or 0)	w_{1N}

$$\text{Error rate} = \text{err}_1 = w_{11}e_{11} + w_{12}e_{12} + \dots + w_{1N}e_{1N}.$$

err_1 is precisely the proportion of times observations are misclassified by the classifier f_1 .

$$\text{Calculate } c_1 = \ln \frac{1 - \text{err}_1}{\text{err}_1}.$$

If $\text{err}_1 = 0$ or $\geq 1/2$, stop. If $\text{err}_1 = 0$, we have hit on the perfect classifier. There is no need to proceed further. Use f_1 . If $\text{err}_1 \geq 1/2$, this is worse than the flipping-coin classification protocol. There is no need to proceed further. Abandon the chosen classification methodology. Try a different one.

Step 2: Calculate a new set of weights.

$$\begin{aligned} w_{21} &= w_{11} \exp(c_1 e_{11}) \\ w_{22} &= w_{12} \exp(c_1 e_{12}) \\ \dots \\ w_{2N} &= w_{1N} \exp(c_1 e_{1N}) \end{aligned}$$

If \mathbf{x}_i is correctly classified, $e_{1i} = 0$ and hence $w_{2i} = w_{1i}$. This means that if an input is correctly classified, the new weight is the same as the old weight. If \mathbf{x}_i is incorrectly classified, $e_{1i} = 1$ and hence $w_{2i} = w_{1i} \left(\frac{1 - \text{err}_1}{\text{err}_1} \right)$. Note that w_{2i} is larger than w_{1i} . If the observation is incorrectly classified, the new weight is larger than the old one.

The new weights are normalized. They are adjusted so that their sum is unity. We use the same notation for the new weights.

Write the data with the new weights.

$$\begin{array}{llll} \text{Sample:} & (\mathbf{x}_1, y_1) & (\mathbf{x}_2, y_2) & \dots & (\mathbf{x}_N, y_N) \\ \text{Weights:} & w_{21} & w_{22} & \dots & w_{2N} (\text{Sum} = 1) \end{array}$$

Note that misclassified observations get more weight!

Use the weighted version of the chosen classification methodology to get a classifier f_2 . What does this mean?

We have a population consisting of N entities: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$. There is a probability distribution $w_{21}, w_{22}, \dots, w_{2N}$ over these entities. Draw a random sample of size N with replacement from this population according to this distribution. Some duplication is possible. If an entity has a large weight, this entity may have many duplicates in the drawn sample. Let the new sample be denoted by

$$(*\mathbf{x}_1, *y_1) \quad (*\mathbf{x}_2, *y_2) \quad \dots \quad (*\mathbf{x}_N, *y_N)$$

Each entity here is one of the original entities. This is the weighted bootstrap sample.

Use the preferred methodology to get a classifier f_2 . Calculate the error rate err_2 associated with f_2 . Use the original training sample not the weighted bootstrap sample.

If $f_2(\mathbf{x}_1) = y_1$ (correct classification), error is zero.

If $f_2(\mathbf{x}_1) \neq y_1$ (wrong classification), error is one.

Etc.

<i>Observation</i>	<i>Error</i>	<i>Weight</i>
(\mathbf{x}_1, y_1)	e_{21} (1 or 0)	w_{21}
(\mathbf{x}_2, y_2)	e_{22} (1 or 0)	w_{22}
...
(\mathbf{x}_N, y_N)	e_{2N} (1 or 0)	w_{2N}

$$\text{Error rate} = \text{err}_2 = w_{21}e_{21} + w_{22}e_{22} + \cdots + w_{2N}e_{2N}.$$

err_2 is precisely the weighted proportion of times observations are misclassified.

$$\text{Calculate } c_2 = \ln \frac{1 - \text{err}_2}{\text{err}_2}.$$

If $\text{err}_2 = 0$, stop, we have hit the perfect classifier. If $\text{err}_2 \geq 1/2$, stop. The chosen methodology of classification is useless.

Step 3: Calculate a new set of weights in exactly the same way the weights are calculated in Step 2 following Step 1.

And so on.

The whole procedure is done some M times. Thus we will have a sequence f_1, f_2, \dots, f_M of classifiers. Thus we have a committee of classifiers. The committee as a whole takes a decision how a new input vector \mathbf{x} , whose group identity is unknown, is to be classified.

Calculate

$$F(\mathbf{x}) = c_1f_1(\mathbf{x}) + c_2f_2(\mathbf{x}) + \cdots + c_mf_m(\mathbf{x}).$$

Classify the object \mathbf{x} into Group 1 if $F(\mathbf{x}) < 0$.

Classify the object \mathbf{x} into Group 2 if $F(\mathbf{x}) \geq 0$.

There is another way the committee can come to a decision: majority voting. Ask each and every committee member how the member will classify the input \mathbf{x} . Determine what the majority decides. Classify \mathbf{x} as per the majority. This, in a gist, is Boosting.

The underlying idea behind Boosting is to combine simple classification rules to form an ensemble rule such that the performance of the single ensemble member is improved, i.e., “boosted.”

5. Do Bagging and Boosting really work?

Does this really work? Does the misclassification rate go down if “Boosting” is employed? There is considerable empirical evidence. [Breiman \(1999b\)](#) reports a number of data sets and compares the performances of Bagging and Boosting with the underlying core methodology of classification being Classification Trees (CART).

Data Set	Size	A	B	Error Rate CART	Error Rate Boost	Error Rate Bag
Heart	1395	2	16	4.9	1.1	2.8
Breast Cancer	699	2	9	5.9	3.2	3.7
Ionosphere	351	2	34	11.2	6.4	7.9
Diabetes	768	2	8	25.3	26.6	23.9
Glass	214	6	9	30.4	22.0	23.2
Soybean	683	19	35	8.6	5.8	6.8
Letters	15,000	26	16	12.4	3.4	6.4
Satellite	4435	6	36	14.8	8.8	10.3
Shuttle	43500	7	9	0.062	0.007	0.014
DNA	2000	3	60	6.2	4.2	5.0
Digit	7291	10	256	27.1	6.2	10.5

Legend: A = #Classes
 B = #Variables (Covariates)

CART: Classification Tree Methodology.

The Classification Tree Methodology was performed on each data once and the corresponding error rate is recorded under CART. For Bagging and Boosting, the CART is the preferred methodology of classification. Note that with the exception of the “Diabetes” data, the error rate decreased considerably.

6. What is a classification tree?

The classification trees methodology was first proposed by Breiman, Friedman, Olshen, and Stone in their monograph published in 1984. This goes by the acronym CART (Classification and Regression Trees). A commercial program called CART can be purchased from Salford Systems. Other more standard statistical softwares such as SPLUS, SPSS, and R package also provide tree construction procedures with user-friendly graphical interface. The package “rpart” in R does classification trees. An excellent documentation on rpart is available on the Internet. Another good source on classification trees is [Zhang and Singer \(2010\)](#).

Let us illustrate the basic ideas of tree construction in the context of a specific example of binary classification. In the construction of a tree, for evaluation purpose, we need the concept of ENTROPY of a probability distribution and Gini’s measure of uncertainty. Suppose we have a random variable X taking finitely many values with some probability distribution.

$$\begin{aligned} X: & \quad 1 \quad 2 \quad \dots \quad m \\ \text{Pr.:} & \quad p_1 \quad p_2 \quad \dots \quad p_m \end{aligned}$$

We want to measure the degree of uncertainty in the distribution (p_1, p_2, \dots, p_m) . For example, suppose $m = 2$. Look at the distributions $(1/2, 1/2)$ and $(0.9, 0.1)$. There is more uncertainty in the first distribution than in the second. Suppose someone is about to crank out or simulate X . It is more comfortable in betting on the outcome of X if the underlying distribution is $(0.9, 0.1)$ than when the distribution is $(1/2, 1/2)$. We want to assign a numerical quantity to measure the degree of uncertainty. Equivalently, we want to measure how chaotic a distribution is. Entropy of a distribution is introduced as a measure of uncertainty.

Entropy $(p_1, p_2, \dots, p_m) = \sum_{i=1}^m -p_i \ln p_i$ = Entropy impurity, with the convention that $0 \ln 0 = 0$.

6.1. Properties

1. $0 \leq \text{Entropy} \leq \ln m$.
2. The minimum 0 is attained for each of the distributions $(1, 0, 0, \dots, 0)$, $(0, 1, 0, \dots, 0)$, \dots , $(0, 0, \dots, 0, 1)$. For each of these distributions, there is no uncertainty. The entropy is zero.
3. The maximum $\ln m$ is attained at the distribution $(1/m, 1/m, \dots, 1/m)$. The uniform distribution is the most chaotic. Under this uniform distribution, the uncertainty is maximum.

There are other measures of uncertainty available in the literature.

Gini's measure of uncertainty for the distribution $(p_1, p_2, \dots, p_m) = \sum_{i \neq j} p_i p_j$.

6.2. Properties

1. $0 \leq \text{Gini's measure} \leq (m - 1)/m$.
2. The minimum 0 is attained for each of the distributions $(1, 0, 0, \dots, 0)$, $(0, 1, 0, \dots, 0)$, \dots , $(0, 0, \dots, 0, 1)$. For each of these distributions, there is no uncertainty. The Gini's measure is zero.
3. The maximum $(m - 1)/m$ is attained at the distribution $(1/m, 1/m, \dots, 1/m)$. Under this uniform distribution, the uncertainty is maximum.

Another measure of uncertainty is defined by $\min\{p_1, p_2, \dots, p_m\}$.

The illustrative example comes from the Yale Pregnancy Outcome Study, a project funded by the National Institute of Health. For details, see [Zhang and Singer \(2010\)](#). The study subjects were women who made a first prenatal visit to a private obstetrics or midwife practice, health maintenance organization, or hospital clinic in the Greater New Haven, Connecticut, area between March 12, 1980 and March 12, 1982. From these women, select those women whose pregnancies ended in a singleton live birth. Each such woman is the sampling unit of the study. The outcome or response variable is: Is this a preterm delivery or a normal time delivery?

At the time of prenatal visit, measurements on 15 variables were collected.

Variable name		Label type	Range/levels
Maternal age	X ₁	Continuous	13–46
Marital status	X ₂	Nominal	Currently married, divorced, separated, widowed, never married
Race	X ₃	Nominal	White, Black, Hispanic, others
Asian, Marijuana use	X ₄	Nominal	Yes, no
Times of using Marijuana	X ₅	Ordinal	≥5, 3–4, 2, 1 (daily), 4–6, 1–3 (weekly), 2–3, 1, <1 (monthly)
Yrs of education	X ₆	Continuous	4–27
Employment	X ₇	Nominal	Yes, no
Smoker	X ₈	Nominal	Yes, no
Cigs smoked	X ₉	Continuous	0–66
Passive smoking	X ₁₀	Nominal	Yes, no
Gravidity hormones/DES	X ₁₁	Ordinal	1–10
Used by mother	X ₁₂	Nominal	None, hormones, DES, both, uncertain
Alcohol (oz/day)	X ₁₃	Ordinal	0–3
Caffeine (mg)	X ₁₄	Continuous	12.6–1273
Parity	X ₁₅	Ordinal	0–7

The training sample consisted of 3861 pregnant women. The objective of the study is to predict whether or not the delivery will be preterm based on the measurements collected at the time of prenatal visit. This is a binary classification problem.

A tree will consist of a root node, internal (circle) nodes, and terminal (box) nodes. Identify each woman in the sample who had a preterm delivery with 0 and who had a normal term delivery with 1.

Step 1: Stuff the root node with all these women.

Step 2: We will create two daughter nodes (left daughter node and right daughter node) out of the root node. Every woman in the root node has to go either to the left daughter node or right daughter node. In other words, we will split the women in the root node into two groups. The splitting is done using one of the predictors. Suppose we start with X₁. In the sample, we have women representing every age from 13 to 43. We need to choose one of the ages for the split. We may decide to split the root node according to the following criterion, for example.

Put a woman in the left daughter node if her age X₁ ≤ 13 years. Otherwise, put the woman in the right daughter node. According to this criterion, some women in the root node go into the left daughter node and the rest go into the right daughter node.

We could split the root node using a different age. For example, put a woman in the left daughter node if her age $X_1 \leq 35$ years. Otherwise, put the woman in the right daughter node.

There are 31 different ways of splitting the root node! We want to choose a good split. The objective is to channel as many women with label 1 as possible into one node and channel as many women with label 0 into the other node. Let us assess the situation when the split was done based on the age 35 years. The composition of the daughter nodes can be summarized by the following 2×2 contingency table.

	Term	Preterm	Total
Left node	3521	198	3719
Right node	135	7	142
Total	3656	205	3861

An interpretation of the table is in order. There were a total of 3656 women who had term (normal) delivery and 205 women who had preterm delivery. Out of 3656 women, 3521 had age ≤ 35 years and 135 were older than 35 years. Out of 205 women, 198 were ≤ 35 years and 7 older than 35 years.

These numbers can be couched in the language of a probability distribution. The left node has a proportion of 3521/3719 ones (term deliveries) and 198/3719 zeros (preterm deliveries). The entropy impurity of the distribution (3521/3719, 198/3719) is given by.

$$-(3521/3719) \ln(3521/3719) - (198/3719) \ln(198/3719) = 0.2079.$$

Similarly, the entropy impurity of the right node is 0.1964. The goodness of the split is defined by.

$$\begin{aligned} &\text{Impurity of the mother node} - \Pr(\text{left node}) * \text{impurity of the left node} \\ &\quad - \Pr(\text{right node}) * \text{impurity of the right node}, \end{aligned}$$

where $\Pr(\text{left node})$ and $\Pr(\text{right node})$ are the probabilities that a subject from the root node falls into the left node and right node, respectively. We can take these probabilities to be 3719/3861 and 142/3861, respectively. For the impurity of the root (mother) node, the probability distribution of 1s and 0s is given by 3656/3861 and 205/3861, respectively. Its entropy is 0.20753. Therefore, the goodness of the split = $0.20753 - (3719/3861) * 0.2079 - (142/3861) * 0.1964 = 0.00001$.

Essentially, the goodness of the split is root node's impurity minus a weighted sum of daughters' impurities. We are shooting for a high value for the goodness of split. Thus for every possible choice of age for a split, we can measure its goodness of split. The optimality principle is choosing that age for which the goodness of split is maximum.

The goodness of allowable age splits

Split value	Impurity of the split		1000 * Goodness of the split
	Left node	Right node	
13	0.00000	0.20757	0.01
14	0.00000	0.20793	0.14
15	0.31969	0.20615	0.17
...
24	0.25747	0.18195	1.50
...
43	0.20757	0.00000	0.01

At the age 24 years, we have an optimal split. Here, we started with age to split the root node. Why age? It could have been any other predictor. The strategy now is clear.

Find the best split and the corresponding impurity reduction for every predictor. Choose that predictor for which impurity reduction is the largest. This is the variable we start with to split the root node.

We need another notion. In this example, The size of the root node is 3861, and the sizes of left daughter node and right daughter node are 3719 and 142, respectively.

After splitting the root node, we look at the left daughter node and right daughter node. We now split the left daughter node into two nodes: left grand daughter node and right grand daughter node. We choose a predictor for the split. Suppose we choose Race for the split. Note that the Race is a nominal variable with five possible values. There are $2^4 - 1$ ways we can split. The possibilities are listed below:

Left grand daughter node	Right grand daughter node
White	Black, Hispanic, Asian, others
Black	White, Hispanic, Asian, others
Hispanic	White, Black, Asian, others
Asian	White, Black, Asian, others
Others	White, Black, Hispanic, Asian
White, Black	Hispanic, Asian, others
White, Hispanic	Black, Asian, others
White, Asian	Black, Hispanic, others
White, others	Black, Hispanic, Asian
Black, Hispanic	White, Asian, others
Black, Asian	White, Hispanic, others
Black, others	White, Hispanic, Asian
Hispanic, Asian	White, Black, others
Hispanic, others	White, Black, Asian
Asian, others	White, Black, Hispanic

Let us look at the split based on White on one hand and Black, Hispanic, Asian, others on the other hand. Channel all women in the left daughter node into left grand

daughter node if she is white. Otherwise, she goes into the right grand daughter node. We can assess how good the split is just the same way as we did earlier.

Thus the splitting goes on using all the predictors at every stage. At every stage, the best predictor with the corresponding threshold split is chosen. This splitting could go on for ever unless we spell out when to stop (pruning strategy).

When do we create a terminal node? We stop splitting a node when its size is smaller than the minimum stipulated (pruning strategy). For example, one could stipulate that if the size of a node is less than 1% of the total sample size, stop splitting. The choice of the minimum size depends on the investigator's perception of utility of the tree.

We will now describe the use of the “rpart” package in R. Terry Therneau and Elizabeth Atkinson (Mayo Foundation) have developed “rpart” (recursive partitioning) package to implement classification trees and regression trees. The method depends what kind of response variable we do have.

```
Categorical → method = "class"
Continuous → method = "anova"
Count      → method = "poisson"
Survival   → method = "exp"
```

They have two monographs on their package with the same title available on the Internet; an introduction to Recursive Partitioning using the RPART routines, February, 2000 and September, 1997. Both are very informative.

Let us illustrate “rpart” command in the context of a binary classification problem. Four data sets are available in the package. Each R command is preceded by a comment in italics.

Download “rpart” from the R source.

Activate “rpart.”

- library(rpart)

Check what data sets are available in the package.

- data(package = “rpart”)

Data sets in package “rpart”:

car.test.frame	Automobile data from “Consumer Reports” 1990
cu.summary	Automobile data from “Consumer Reports” 1990
kyphosis	Data on children who have had corrective spinal surgery
solder	Soldering of components on printed-circuit boards

Let us look at “kyphosis” data.

> data(kyphosis)

Check how large the data set is

```
> dim(kyphosis)
```

```
[1] 81 4
```

Look at the top six rows of the data.

```
> head(kyphosis)
```

	Kyphosis	Age	Number	Start
1	absent	71	3	5
2	absent	158	3	14
3	present	128	4	5
4	absent	2	5	1
5	absent	1	4	15
6	absent	1	2	16

Look at the documentation on the data.

```
< ?kyphosis
```

There are 81 children on whom a surgery was done on the spine. For each child, after surgery Kyphosis (spinal deformity) was determined. This is our response variable, which is binary. There are three predictors in the study. The age of child in months at the time of surgery is recorded. The variable “Number” identifies the number of vertebrae which are misaligned. The vertebrae are numbered serially from the top to the bottom with numbers 1 to total number of vertebrae. The surgery was done only on one of the misaligned vertebrae. The topmost misaligned vertebra was chosen for surgery. This is recorded under the variable’s name “Start.”

Look at the documentation on “rpart.” If we let the partition continue without any break, we will end up with a saturated tree. Every terminal node is pure. It is quite possible some terminal nodes contain only one data point. We need to declare when the splitting has to stop. If a node is not split further, it will be a terminal node. Each terminal node will then have some children, some of them will have Kyphosis absent and the rest Kyphosis present. Label the terminal node with “Kyphosis present” if the majority of children in the node have Kyphosis present. Otherwise, the node is labeled with “Kyphosis absent.”

We need to arrest the growth of the tree. One possibility is to demand that if a node contains 20 observations or less no more splitting is to be done at this node. This is the default setting in “rpart.” The norm is if the size of the node is 20% of the sample size or less, splitting has to be stopped. *Let us execute the R command rpart on kyphosis data.*

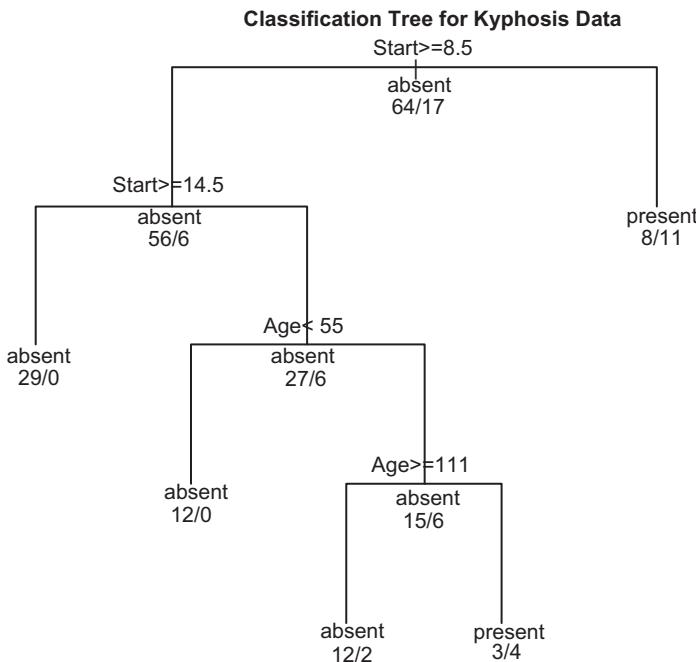
```
< MB <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
```

The output folder “MB” will explain how the tree is created. We need to get the actual tree. The following commands provide a tree.

```
> plot(MB, uniform = T, margin = 0.1)
```

```
> text(MB, use.n = T, all = T)
```

```
> title(main = “Classification Tree for Kyphosis Data”)
```



Comments

- The root node has 81 children with 64 having Kyphosis absent and 17 Kyphosis present. The majority of children have Kyphosis absent. This is noted in the root node.
- The root node is split. The best predictor is Start and the best cut-point is 8.5. The choice is made according to the entropy criterion. If a child has $\text{Start} \geq 8.5$, the child goes into the left node. Otherwise, the child goes into the right node.
- The right node has 19 children with 11 of them having Kyphosis absent and eight of them Kyphosis present. The node is not split further. Its size 19 is ≤ 20 , the default setting. The majority of children in this node had Kyphosis present. This is a terminal node and labeled with Kyphosis present.
- The left node has 62 children with 56 of them having Kyphosis absent and six Kyphosis present. This node is split. The best predictor is Start and the optimal cut-point is 14.5. If a child in this node has $\text{Start} \geq 14.5$, the child will go into the left node. Otherwise, the child will go into the right node. Splitting continues until the size is ≤ 20 or the node is pure, i.e., every child has the same label.
- This tree produced five terminal nodes. Three of the nodes are labeled Kyphosis absent and the remaining Kyphosis present. Two of the nodes are pure.

The classification tree produced is used for classifying a child. Suppose we have all covariate information on the child and we want to predict whether or not Kyphosis will be absent after surgery. We can feed the data into the root node. The child will land into one of the terminal nodes. Look at the label of the terminal node. The label is assigned to the child. A verbal description of the classification tree is provided.

- If a child has $\text{Start} < 8.5$, predict that Kyphosis will be present.

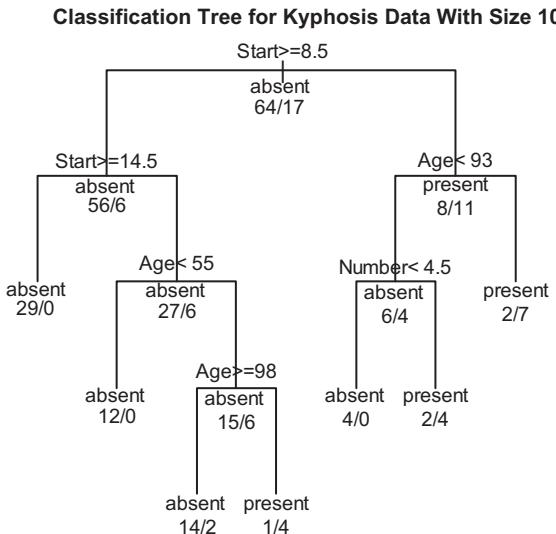
- b. If a child has $14.5 \leq \text{Start}$, predict that Kyphosis will be absent.
- c. If a child has $8.5 \leq \text{Start} < 14.5$ and $\text{Age} < 55$ months, predict that Kyphosis will be absent.
- d. If a child has $8.5 \leq \text{Start} < 14.5$ and $\text{Age} \geq 111$ months, predict that Kyphosis will be absent.
- e. If a child has $8.5 \leq \text{Start} < 14.5$ and $55 \leq \text{Age} < 111$ months, predict that Kyphosis will be present.

The tree is very revealing. The covariate “Number” has no role in the classification. One can view the classification tree as non-parametric regression with response variable being binary. The tree is searching for patterns in the data empirically. For example, it is observed that there are 29 children with all of them having $\text{Start} \geq 14.5$ and Kyphosis absent. If the vertebra number is 15 or higher, Kyphosis seems to be absent after surgery. Surgery on the lower part of the spine seems to lead to Kyphosis absent. Now we need to judge how good the tree is.

We have 81 children in our study. We know for each child whether Kyphosis is present or absent. Pour the data on the covariates of a child into the root node. See which terminal node the child settles in. Classify the child accordingly. We know the true status of the child. Note down whether or not a mismatch occurred. Find the total number of mismatches. The misclassification rate can be calculated. Add up all the minority numbers in the terminal nodes.

$$\begin{aligned}\text{Misclassification rate} &= \text{re-substitution error} \\ &= 100 * (8 + 0 + 2 + 3) / 81 = 16\%.\end{aligned}$$

Is this acceptable? We can increase the size of the tree by reducing the threshold number 20. Let us do it. Let us the threshold size to 10. This may lead to a reduction in misclassification rate.



The following are the relevant commands.

```
> MB <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis, control =
rpart.control(minsplit = 10))
> plot(MB, margin = 0.1, uniform = T)
> text(MB, use.n = T, all = T)
> title(main = "Classification Tree for Kyphosis Data With Size 10")
```

The input “control = rpart.control(minsplit = 10)” in the MB folder lets us to incorporate the minimum size set at 10. The tree is different. We now have seven terminal nodes. The misclassification rate is given by

$$\text{Misclassification rate} = [(0 + 0 + 2 + 1 + 0 + 2 + 2)/81] * 100 = 8.6\%.$$

The misclassification rate went down by about 50%.

How do we arrest the growth of a tree? Are there alternative ways?

One method is to introduce an optimality criterion. One optimality criterion is the re-substitution error. In principle how this works. Take any tree T with its terminal nodes. Pour the data through the root node. See in which terminal node each observation lands. Find out how many of these observations are misclassified. The proportion of misclassified observations is called the re-substitution error. We could set up a goal. Find that tree for which the re-substitution error is minimum.

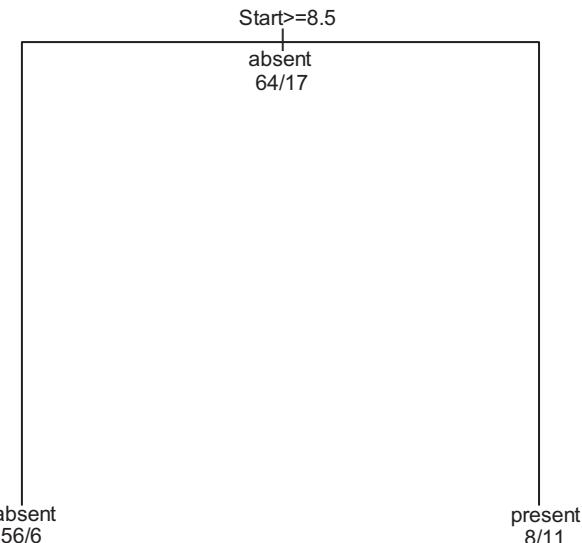
This criterion is not good. The saturated tree will be the answer. Introduce a complexity parameter (cp) $0 \leq \alpha < 1$. Let T be any tree. Define $R(T) = R^{\text{resub}}(T) + \alpha(|T|)$.

1. For any given tree T , one can calculate the re-substitution error $R^{\text{resub}}(T)$. The symbol T stands for the number of terminal nodes of T . The entity α is the penalty for having too many terminal nodes.
2. Find a tree for which $R(T)$ is minimum. The saturated tree will lose out here. For the saturated tree, $R^{\text{resub}}(T)$ is zero. But $\alpha|T|$ will be very large.
3. This criterion of optimization is similar to AIC (Akaike Information Criterion) in spirit in model selection problems of regression.

In rpart, one can set cp and ask for an optimal tree. Let us do this. The relevant code is:

```
> MB <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis, control =
rpart.control(cp = 0.05))
> plot(MB, uniform = T, margin = 0.1)
> text(MB, use.n = T, all = T)
> title(main = "Classification Tree for KYphosis Data with CP = 0.05")
```

Classification Tree for KYphosis Data with CP = 0.05



Defaults in the command

1. For each covariate, the best split is decided based on Gini's index.
2. If one wants to use "entropy" to decide the best split, use the input: split = "information".
3. One can inject Bayesian paradigm into classification. The default priors are proportional to the data counts. If you need different priors, use the input: parms = list(prior = c(0.65, 0.35)), for example.

If the complexity parameter (cp) is set at 0.05, the tree has only two terminal nodes. Its misclassification rate is given by $[(6 + 8)/81] * 100 = 17.3\%$. This is not much different from the one when the minimum size is set at 20. In the tree that is produced, only one covariate, namely Start, plays a role.

7. Classification tree versus logistic regression

A classification tree is an empirical summary of the data. We cannot answer questions as to the significance of the covariates. The logistic regression is better equipped to tackle issues on significance. We can fit a logistic regression model to the kyphosis data. The model is given by

$$\Pr(\text{present}) = \frac{\exp(\beta_0 + \beta_1 * \text{Age} + \beta_2 * \text{Number} + \beta_3 * \text{Start})}{1 + \exp(\beta_0 + \beta_1 * \text{Age} + \beta_2 * \text{Number} + \beta_3 * \text{Start})}$$

$$\Pr(\text{absent}) = \frac{1}{1 + \exp(\beta_0 + \beta_1 * \text{Age} + \beta_2 * \text{Number} + \beta_3 * \text{Start})}$$

for some parameters $\beta_0, \beta_1, \beta_2$, and β_3 . The last three parameters are the regression coefficients associated with the predictors. The model is fitted using the "glm" command in R.

```
> MB1 <- glm(Kyphosis ~ Age + Number + Start, data = kyphosis, family = binomial)
> summary(MB1)
```

The output is given by

Call:

```
glm(formula = Kyphosis ~ Age + Number + Start, family = binomial, data = kyphosis)
```

Deviance residuals:

Min	1Q	Median	3Q	Max
-2.3124	-0.5484	-0.3632	-0.1659	2.1613

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-2.036934	1.449575	-1.405	0.15996
Age	0.010930	0.006446	1.696	0.08996
Number	0.410601	0.224861	1.826	0.06785
Start	-0.206510	0.067699	-3.050	0.002297
---				**

Signif. codes: 0 "***" 0.001 "**" 0.01 "*" 0.05 "." 0.1 " " 1

Null deviance: 83.234 on 80 degrees of freedom

Residual deviance: 61.380 on 77 degrees of freedom

AIC: 69.38

Interpretation of the output

The estimates of the unknown parameters are given by:

$$\hat{\beta}_0 = -2.04$$

$$\hat{\beta}_1 = 0.01$$

$$\hat{\beta}_2 = 0.41$$

$$\hat{\beta}_3 = -0.21$$

The covariate Start is very significant with p -value 0.002. The other covariates are not significant at 5% level of significance but significant at 10% level. The model fit is good with residual deviance at 61.38 giving a ratio $61.38/77 = 0.8$, which is well-below unity.

The output is somewhat in agreement with that of the classification tree. We have noted that in the classification tree, only two variables Start and Age played a role in the build-up of the tree.

8. Random forest

A random forest is a modification (extension?) of bagging. In bagging, one can use any regression or classification method as the basic tool. However, a random forest uses only classification or regression trees as the underlying method.

The *modus operandi* of a random forest runs as follows. Suppose N is the size of the training sample. For illustration, let us assume that we have a classification problem with input vector x consisting of 10 covariates or predictors, say.

Step 1: Select a random sample of size N with replacement from the training data. All these data go into the root node. We need to split the node into two daughter nodes.

Step 2: Choose and fix an integer $1 < m < 10$. Select a random sample of m predictors without replacement from the set of 10 predictors on hand. From this sample of m predictors, choose the best predictor to split the root node into two daughter nodes in the usual way either using entropy or Gini's index in impurity calculations.

Step 3: We need to split the daughter nodes. Take the “left daughter node.” Select a random sample of m predictors without replacement from the set of all predictors. From this sample of m predictors, choose the best predictor to split the node. For the “right daughter node,” select a random sample of m predictors without replacement from the set of all predictors. From this sample of m predictors, choose the best predictor to split the node.

Step 4: We will now have four grand daughters. Split each of these nodes the same way as outlined in Step 2 or 3.

Step 5: Continue this process until we are left all with pure nodes. No pruning is done. We will have a very large tree.

Step 6: Repeat Steps 1–5, 500 times. Thus we will have 500 trees.

This is the so-called random forest.

How do we classify a new data vector x ? Feed the vector into each of the trees. Find out what the majority of these trees say. The vector x is classified accordingly.

This is essentially “bagging” procedure with randomization on predictors introduced at every node for splitting!

Let us look at the package “randomForest.” Download this package and make it active. Look at its documentation.

- ?randomForest

8.1. A discussion of inputs

1. The first input x is the matrix of data on predictors. The second input is the corresponding data on the response variable (class or quantitative). In lieu of these inputs, one can give a regression formula.
2. `ntree` = number of trees. The default is set at 500.
3. `mtry` = fixed number of predictors to be selected randomly at every node. The default is set at the integer part of the square root of the number of predictors.
4. `importance`: with respect to every category in the response one gets information on how important the predictors in classification. `Importance = F` is the default.

5. proximity: the command calculates proximity scores between any two rows. This is essentially the distance between any two subjects in the sample based on the covariates. The result is an $N \times N$ matrix.

Let us apply the command on the “iris” data.

Download the data.

```
> data(iris)
```

Understand the data.

```
> head(iris)
```

	Sepal. Length	Sepal. Width	Petal. Length	Petal. Width	Species
1	5.1	3.5	1.4	0.0	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

```
> dim(iris)
```

```
[1] 150 5
```

```
> sapply(iris, class)
```

```
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
"numeric"     "numeric"    "numeric"    "numeric"    "factor"
```

```
> table(iris$Species)
```

setosa	versicolor	virginica
50	50	50

Data are obtained on three species of flowers: setosa, versicolor, and virginica. Four measurements are made on each flower: Sepal length, Sepal width, Petal length, and Petal width. Data are obtained on 50 flowers of each species. The response variable is Species, which is a class variable with three levels. This is a classification problem with four covariates.

Let us invoke the command.

```
> MB <- randomForest(Species ~ ., data = iris, importance = T, proximity = T)
```

Look at the output.

```
> print(MB)
```

Call:

randomForest(formula = Species ~ ., data = iris, importance = T, proximity = T)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 4%				
Confusion matrix:				
	setosa	versicolor	virginica	class.error
setosa	50	0	0	0.00
versicolor	0	47	3	0.06
virginica	0	3	47	0.06

Understand the output.

1. R says that this is a classification problem.
2. R produces a forest of 500 trees.
3. There are four predictors. At every node, it chose two predictors at random. In our notation $m = 2$.
4. The given data were poured into the root node. The acronym “OOB” stands for “out-of-bag.” The OOB (error) estimate is 4%. The confusion matrix spells out where errors of misclassification occurred. All 50 flowers of setosa are correctly classified. Three flowers of versicolor are misclassified. Three flowers of virginica are misclassified.

What else is available in the output folder “MB.”

> names(MB)

```
[1] "call"      "type"      "predicted"   "err.rate"
[5] "confusion"  "votes"     "oob.times"   "classes"
[9] "importance" "importanceSD" "localImportance" "proximity"
[13] "ntree"     "mtry"      "forest"     "y"
[17] "test"      "inbag"     "terms"
```

Let us look at “importance.”

> round(importance(MB), 2)

	setosa	versicolor	virginica	Mean Decrease	Mean Decrease Gini
Sepal.Length	1.54	1.70	1.75	1.35	9.61
Sepal.Width	1.04	0.30	1.19	0.72	2.28
Petal.Length	3.62	4.40	4.18	2.48	41.68
Petal.Width	3.88	4.43	4.26	2.52	45.68

Interpretation of the output

1. For setosa, the most important variable that played a role in the classification is Petal.Width followed by Petal.Length. The same story holds true for every other flower species.
2. Since Petal.Length and Petal.Width are consistently important for every flower, it means for classification purpose, we could use only these two variables. We will attempt this exercise later.

We have 500 trees sitting inside inner recesses of the computer. There is no need to look at them. We can make use of them for classification purpose.

I have a new flower whose identity is unknown. I know its measurements: 6.1, 3.9, 1.5, 0.5. We want to classify this flower. Pour these measurements into the root node of every tree in the forest. See what the majority says. This is a prediction problem. Create a data folder with these measurements.

```
> MB1 <- data.frame(Sepal.Length = 6.1, Sepal.Width = 3.9, Petal.Length = 1.5,
Sepal.Width = 0.5)
```

Look at the output of this data folder.

```
> MB1
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	6.1	3.9	1.5	0.5

Predict the species.

```
> MB2 <- predict(MB, newdata = MB1, type = "class")
```

Output

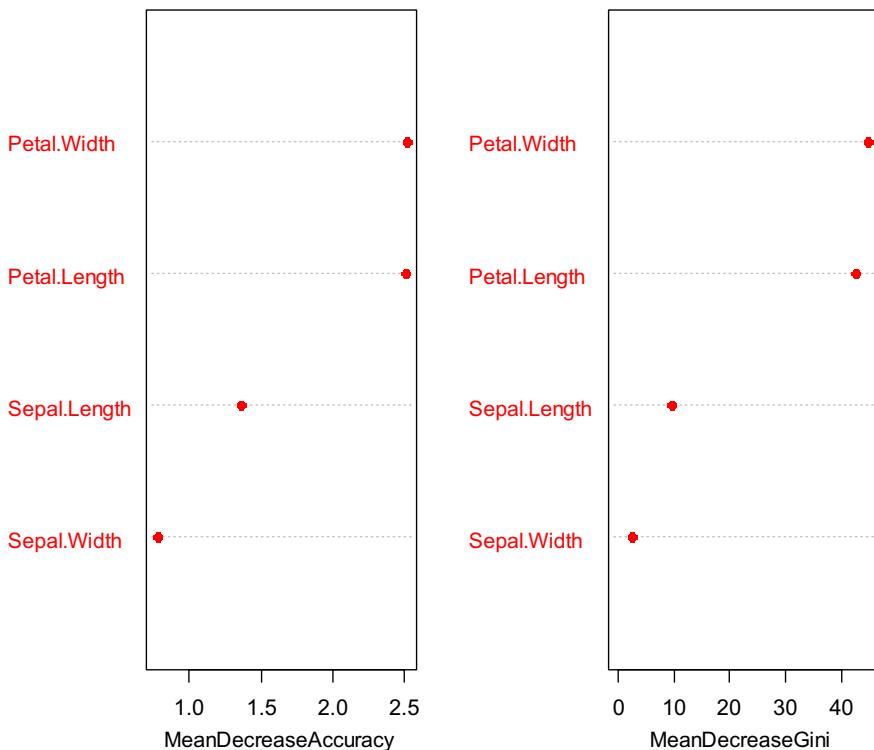
```
> MB2
[1] setosa
Levels: setosa versicolor virginica
```

This flower is classified as setosa.

How important the covariates are in the classification problem? One can get a graph. The following is the relevant R code.

```
> varImpPlot(MB, pch = 16, col = "red", n.var = 4, sort = T,
main = "Importance of Variables for the Iris data")
```

Importance of Variables for the Iris data



How importance is measured?

There are 500 trees in the background. We want to measure the importance of one of the covariates X_1 , say. In our example, X_1 could be any one of the four covariates. Choose and fix one of the trees T in the forest. Note that there is a bootstrap sample behind it. Calculate the out-of-bag (OOB) error estimate, $P_T(\text{OOB})$. Take a random permutation of X_1 -values in the bootstrap sample. Construct a tree and calculate the corresponding OOB error estimate, $P_T(\text{OOB}_1)$. A raw score for X_1 is calculated by

$$\text{Raw}_T(X_1) = P_T(\text{OOB}_1) - P_T(\text{OOB}).$$

The philosophy behind what we are doing is that if X_1 is important permuting the X_1 -values in the bootstrap sample will result a high value for the OOB error estimate, $P_T(\text{OOB}_1)$. Then we would expect a large value for the raw score $\text{Raw}_T(X_1)$. Calculate the raw scores for every tree in the forest. We will then have 500 such numbers. Calculate their average and standard deviation. The z -score, average/standard deviation, is the so-called mean decrease in accuracy for the X_1 -covariate. This is what the importance graph depicts for each covariate. Intuitive explanation is that if X_1 is important one would expect large raw scores and a high z -score. If X_1 is not important, one would expect a small z -score.

There is another way to measure the importance of a covariate X_1 . Take a tree T in the forest. Note that the Gini impurity index for any parent node is larger than the value of that measure for its two daughter nodes collectively. The difference between the parent's impurity index and its daughters' collective impurity index is averaged (G_1) over all parent nodes and trees. Now permute the X_1 -values randomly in each bootstrap sample. Calculate the difference between the parent's impurity index and its daughters' collective impurity index for each parent node. The differences are averaged (G_2) over all parent nodes and trees. The difference $G_2 - G_1$ is the Mean Decrease according to Gini. This is also reported in the importance graph.

As we can see from the importance graph, the petal length and petal width are the most important predictors of the identity of species. Perhaps a random forest built on these two covariates should be adequate for classification. Let us work out the details.

```
> MB <- randomForest(Species ~ Petal.Length + Petal.Width, data = iris,
importance = T, proximity = F)
```

```
> print(MB)
```

Call:

```
randomForest(formula = Species ~ Petal.Length + Petal.Width, data = iris,
importance = T, proximity = F)
```

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 1

OOB estimate of error rate: 4.67%

Confusion matrix:

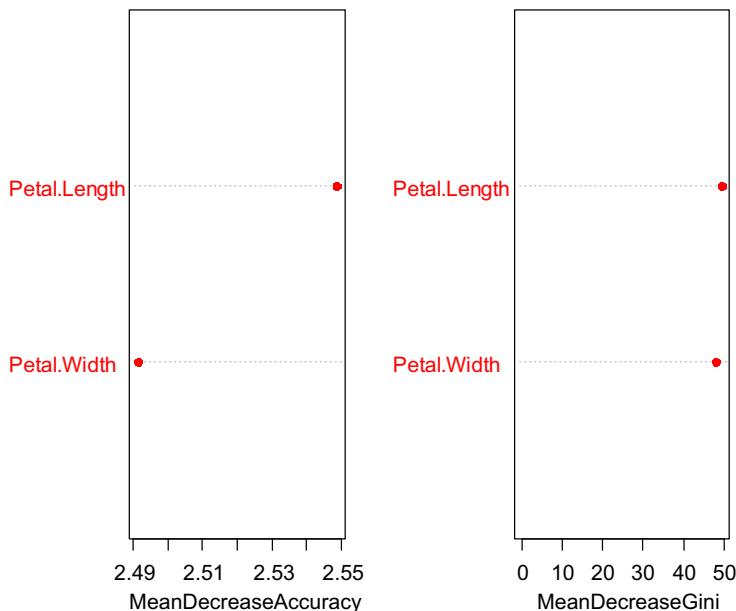
	setosa	versicolor	virginica	class.error
setosa	50	0	0	0.00
versicolor	0	47	3	0.06
virginica	0	4	46	0.08

```
> round(importance(MB), 2)
```

	setosa	versicolor	virginica	Mean Decrease Accuracy	Mean Decrease Gini
Petal. Length	3.82	4.61	4.24	2.55	49.45
Petal. Width	3.76	4.41	4.30	2.49	47.87

```
> varImpPlot(MB, pch = 16, col = "red", n.var = 2, sort = T, main =
"Importance of Variables for the Iris data based on Petal Measurements")
```

**Importance of
Variables for the Iris data based on Petal Measurements**



Interpretation of the output

1. The out-of-bag error estimate has gone up from 4% to 4.67%. Four flowers of virginica are misclassified into versicolor. This is the only difference between the random forest based on the four covariates and the one based on just two covariates.
2. As for importance, there seems to be no difference between Petal Length and Petal Width.

9. Random forest, genetics, and cross-validation

We want to illustrate usefulness of random forests in statistical genetics. We have data from NIOSH (The National Institute of Occupational Safety and Health), Morgantown, West Virginia. The subjects of the study come from a number of work environments such as sawmills, steel foundries, etc. The subjects are divided into two groups: those who are diagnosed to have occupational asthma (cases) (Group 2) and those who do not have asthma (controls) (Group 5). Blood samples are drawn from the participants of the study and data on more than 2000 SNPs (Single Nucleotide Polymorphisms) are collected. A genome wide association analysis is carried out and seven SNPs stood out to be significant after the Bonferroni adjustment. Measurements on eight demographic variables are collected on each

participant. The data are not yet published. The purpose of this section is to explain what we did with the data and how random forests helped us to understand the underlying factors of occupational asthma. We also want to introduce ideas on cross-validation in data analysis. The data were in EXCEL format. The data were copied onto a clipboard.

Input the data into R from my flash.

```
> MB <- read.delim("clipboard")
```

Look at the top six rows of the data.

```
> head(MB)
```

	Sampleid	GroupC	rs1264457	rs1573294	rs1811197	rs3128935	rs4711213	rs7773955
1	CAN235	2	1	1	2	1	3	1
2	CAN237	2	1	1	3	1	3	1
3	CAN242	2	1	3	3	1	3	1
4	CAN260	2	1	2	3	2	3	1
5	CAN263	2	1	2	3	1	3	1
6	CIN309	2	1	1	2	1	3	1

	rs928976	Sex	Age	Height	ExpMonths	Atopy1	Smoking	PackYears
1	2	M	30.00000	173	172.0	pos	ex	6.0
2	1	M	23.00000	184	7.0	neg	current	4.0
3	1	M	19.00000	175	13.0	pos	never	0.0
4	2	M	38.00000	178	34.7	pos	ex	5.0
5	1	M	58.00000	170	346.4	pos	ex	31.3
6	3	M	26.16389	173	76.8	pos	never	0.0

The first column is the serial number that comes from EXCEL. This is not part of the data. The second column is the subject's ID. This is also not part of the data. The third column identifies to which group the subject belongs (2: case and 5: control). The next seven columns give SNP data. Each SNP is a categorical variable with three levels corresponding three genotypes. The genotypes are codified as 1, 2, and 3. The last seven variables pertain to demographic data and work-related data. The covariate "ExpMonths" records how long the subject has been working in industry. The covariate "Pack years" records how long the subject has been smoking combined with number of packs smoked. The goal of the study is to identify genetic factors and demographic factors that lead to occupational asthma.

Check the dimension of the data.

```
> dim(MB)
```

```
[1] 159 16
```

Response variable: GroupC	
<i>Covariates</i>	
SNPs:	rs1264457 rs1573294 rs1811197 rs3128935 rs4711213 rs7773955 rs928976
Demographics:	Sex Age Height ExpMonths Atopy1 Smoking PackYears

Remove the individuals on whom demographics are missing. Remove the id too.

```
> MB1 <- MB[-c(81, 87, 95, 133, 146, 149), -1]
```

Effective size of the data

```
> dim(MB1)  
[1] 153 15
```

What kind of variable is GroupC?

```
> class(MB1[, 1])  
[1] "numeric"
```

Convert that into a categorical variable.

```
> MB1[, 1] <- as.factor(MB1[, 1])  
> class(MB1[, 1])  
[1] "factor"
```

The goal is cross-validation. We want to set aside 1/3rd of the data randomly selected. Perform random forest methodology on the remaining 2/3rd of the data.

Sample size

```
> n <- dim(MB1)[1]  
> n  
[1] 153
```

Sample size for implementation of random forest methodology.

```
> k <-(2/3) * n  
> k  
[1] 102
```

Select a random sample of 102 numbers from 1 to 153.

```
> s <- sample(1:n, k)
> s
```

```
[1] 54 20 89 49 15 135 82 76 4 113 136 22 125 59 142 42 63 144
[19] 83 130 35 146 61 116 78 16 74 44 123 106 81 140 80 52 12 55
[37] 68 112 117 88 134 126 53 138 11 151 67 28 107 60 40 109 95 121
[55] 72 104 29 31 128 79 43 98 65 10 86 108 87 105 77 101 129 18
[73] 30 1 19 90 58 69 57 71 149 124 26 115 66 93 152 21 102 137
[91] 120 70 41 32 119 131 143 122 47 150 36 51
```

Create the learning set of 102 individuals.

```
> MBLset <- MB1[s, ]
```

Create the test set of 51 individuals.

- MBTset <- MB1[-s,]

The learning set had 51 individuals belonging to Group 2 and 50 belonging to Group 5.

```
> table(MBLset[, 1])
```

```
2      5
52     50
```

Run the random forest method on the learning set.

```
> model.rf <- randomForest(GroupC ~ ., data = MBLset, importance = T,
+ proximity = T)
```

Look at the output.

```
> print(model.rf)
```

Call:

```
randomForest(formula = GroupC ~ .,      data = MBLset,      importance = T,
proximity = T)
```

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 3

OOB estimate of error rate: 10.78%

Confusion matrix:

	2	5	class.error
2	45	7	0.1346154
5	4	46	0.0800000

Comments on the output: Each node is split using three randomly selected covariates. A forest of 500 trees is created. The learning sample was poured into the trees. The majority rule was used. The out-of-bag (OOB) error (misclassifications) was 10.78%. Seven cases have been misclassified as control. Four controls have been misclassified as cases.

Which covariates played an important role in the groups. The importance measure is given below:

```
> round(model.rf$importance, 3)
```

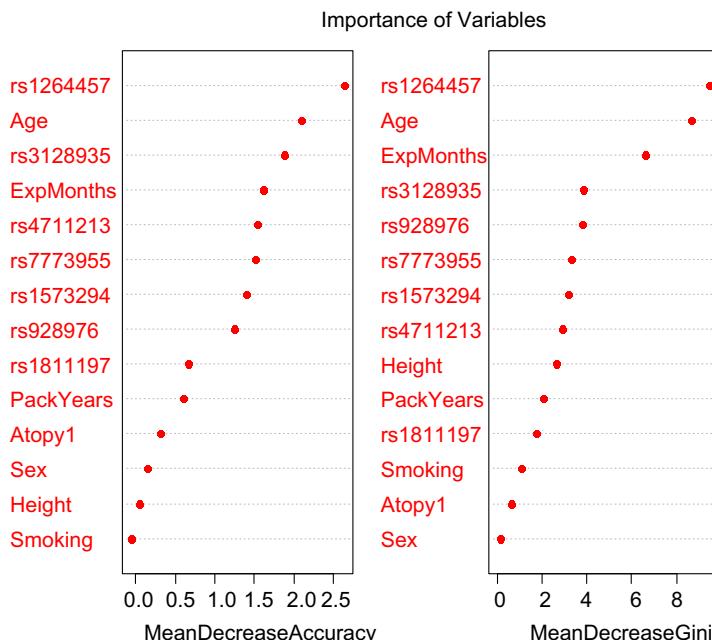
	2	5	MeanDecrease Accuracy	Mean DecreaseGini
rs1264457	0.088	0.101	0.093	9.493
rs1573294	0.007	0.024	0.016	3.156
rs1811197	0.010	0.002	0.006	1.739
rs3128935	0.035	0.017	0.026	3.848
rs4711213	0.025	0.018	0.021	2.949
rs7773955	0.030	0.013	0.022	3.304
rs928976	0.016	0.018	0.017	3.799
Sex	0.000	0.001	0.000	0.183
Age	0.032	0.064	0.046	8.654
Height	-0.002	0.003	0.000	2.656
ExpMonths	0.038	0.021	0.029	6.588
Atopy1	-0.001	0.005	0.002	0.656
Smoking	0.000	-0.001	0.000	1.071
PackYears	0.003	0.006	0.005	2.042

The top five covariates

Group 2: rs1264457; Age; rs3128935; ExpMonths; rs4711213
 Group 5: rs1264457; Age; ExpMonths; rs3128935; rs928976

The corresponding importance graph is given below:

```
> varImpPlot(model.rf, pch = 16, col = "red," sort = T, n.var = 14, main =
  "Importance of Variables")
```



The test set is used for prediction.

```
> predict.rf <- predict(model.rf, newdata = MBTset, type = "class")
```

The accuracy of prediction is determined.

```
> acc.rf <- 100 * sum(predict.rf == MBTset$GroupC)/dim(MBTset)[1]
```

Prediction accuracy is 92.15%.

```
> acc.rf  
[1] 92.15686
```

10. Regression trees

We now focus on developing a regression tree when the response variable is quantitative. Let us work out the build-up of a tree using an example. The data set “bodyfat” is available in the package “mboost.” Download the package and the data. The data has 71 observations on 10 variables. Body fat was measured on 71 healthy German women using Dual Energy X-ray Absorptiometry (DXA). This reference method is very accurate in measuring body fat. However, the setting-up of this instrument requires a lot of effort and is of high cost. Researchers are looking ways to estimate body fat using some anthropometric measurements such as waist circumference, hip circumference, elbow breadth, and knee breadth. The data gives these anthropometric measurements on the women in addition to their age.

Here is the data.

```
> data(bodyfat)
```

Size of the data

```
> dim(bodyfat)  
[1] 71 10
```

Top six rows of the data

```
> head(bodyfat)
```

	age	DEXfat	waistcirc	hipcirc	elbowbreadth	kneebreadth	anthro3a
47	57	41.68	100.0	112.0	7.1	9.4	4.42
48	65	43.29	99.5	116.5	6.5	8.9	4.63
49	59	35.41	96.0	108.5	6.2	8.9	4.12
50	58	22.79	72.0	96.5	6.1	9.2	4.03
51	60	36.42	89.5	100.5	7.1	10.0	4.24
52	61	24.13	83.5	97.0	6.5	8.8	3.55
	anthro3b	anthro3c	anthro4				
47	4.95	4.50	6.13				
48	5.01	4.48	6.37				
49	4.74	4.60	5.82				
50	4.48	3.91	5.66				
51	4.68	4.15	5.91				
52	4.06	3.64	5.14				

Ignore the last four measurements. Each one is a sum of logarithms of three of the four anthropometric measurements. We now want to create a regression tree. As in the construction of a classification tree, all data points go into the root node to begin with. We need to select one of the covariates and a cut-point to split the root node. Let us start with the covariate “waistcirc” and the cut-point 88.4, say. All women with waistcirc <88.4 are shepherded into the left node and others into the right node. Forty of the women move into the left node and the remaining 31 into the right node. We need to judge how good the split is. We use variance as the criterion. Calculate the variance of “DEXfat” of all women in the root node. It is 121.9426. Calculate the variance of “DEXfat” of all women in the left node. It is 33.72712. Calculate the variance of “DEXfat” of all women in the right node. It is 52.07025. The goodness of the split is defined by.

Goodness of the split

$$\begin{aligned} &= \text{Parent node's variance} - \text{Waited sum of daughters' variances} \\ &= 121.9426 - [(40/71) * 33.72712 + (31/71) * 52.07025] = 80.2065. \end{aligned}$$

The goal is to find that cut-point for which the goodness of the split is maximum. Maximization is carried out over all cut-points of a covariate and all covariates. Select the best covariate and the corresponding optimal cut-point to start the tree. Follow the same principle at every stage.

The variances are calculated using R.

```
> var(bodyfat$DEXfat)
[1] 121.9426
```

Identify the data in the left node.

```
> MB1 <- subset(bodyfat, bodyfat$waistcirc < 88.4)
> var(MB1$DEXfat)
[1] 33.72712
> mean(bodyfat$DEXfat)
[1] 30.78282
```

Identify the data in the right node.

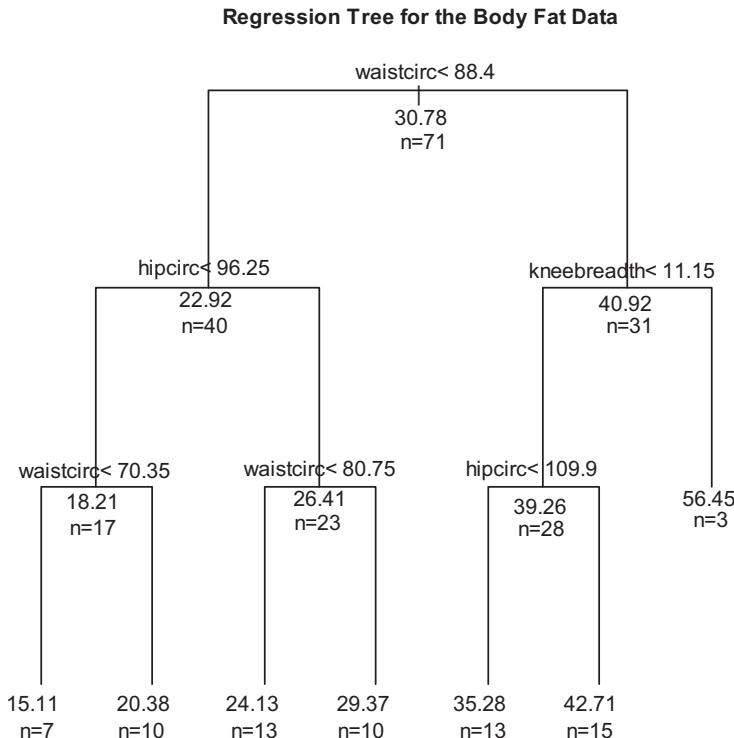
```
> MB2 <- subset(bodyfat, bodyfat$waistcirc >= 88.4)
> dim(MB1)
[1] 40 10
> dim(MB2)
[1] 31 10
> var(MB2$DEXfat)
[1] 52.07205
```

Let us use rpart package to build a regression tree. We need to prune the tree. If the size of a node is 10 or less, do not split the node. We have omitted the covariate age.

```
> MB <- rpart(DEXfat ~ waistcirc + hipcirc + elbowbreadth + kneebreadth,
  data = bodyfat, control = rpart.control(minsplit = 10))
```

Plot the tree with annotation.

```
> plot(MB, uniform = T, margin = 0.1)
> text(MB, use.n = T, all = T)
> title(main = "Regression Tree for the Body Fat Data")
```



Interpretation of the tree

1. At each node, the mean of DEXfat is reported.
2. At each node the size n of the node is reported.
3. The tree has seven terminal nodes.
4. The variable elbowbreadth has no role in the tree.
5. How does one carry out prediction here? Take any woman with given anthropometric measurements. Pour the measurements into the root node. The data will settle into one of the terminal nodes. The mean of the DEXfat reported in the terminal node is the predicted DEXfat for the woman.

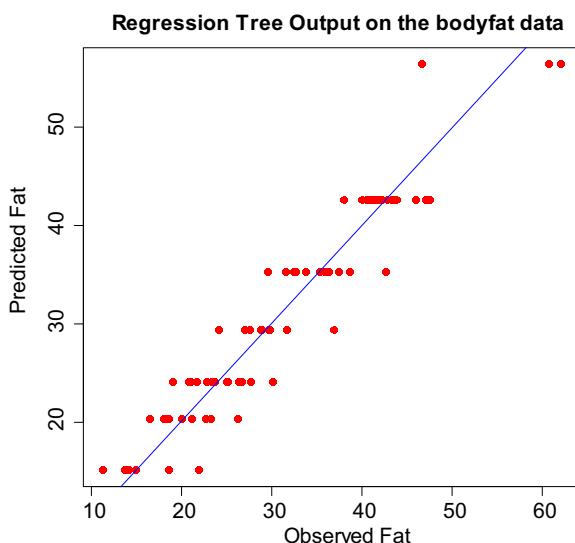
The predicted values as per the tree can be obtained. The observed values and predicted values can be put in the same folder.

```
> MB3 <- predict(MB, newdata = bodyfat)
> MB4 <- data.frame(bodyfat$DEXfat, PredictedValues = MB3)
> MB4
```

	bodyfat.DEXfat	PredictedValues
47	41.68	42.71133
48	43.29	42.71133
49	35.41	35.27846
50	22.79	24.13077
51	36.42	35.27846
52	24.13	29.37200
53	29.83	29.37200
54	35.96	35.27846
55	23.69	24.13077
56	22.71	20.37700
57	23.42	24.13077
58	23.24	20.37700
59	26.25	20.37700
60	21.94	15.10857
61	30.13	24.13077
62	36.31	35.27846
63	27.72	24.13077
64	46.99	42.71133
65	42.01	42.71133
66	18.63	20.37700
67	38.65	35.27846
68	21.20	20.37700
69	35.40	35.27846
70	29.63	35.27846
71	25.16	24.13077
72	31.75	29.37200
73	40.58	42.71133
74	21.69	24.13077
75	46.60	56.44667
76	27.62	29.37200
77	41.30	42.71133
78	42.76	42.71133
79	28.84	29.37200
80	36.88	29.37200
81	25.09	24.13077
82	29.73	29.37200
83	28.92	29.37200
84	43.80	42.71133
85	26.74	24.13077
86	33.79	35.27846
87	62.02	56.44667
88	40.01	42.71133
89	42.72	35.27846
90	32.49	35.27846
91	45.92	42.71133
92	42.23	42.71133

	bodyfat.DEXfat	PredictedValues
93	47.48	42.71133
94	60.72	56.44667
95	32.74	35.27846
96	27.04	29.37200
97	21.07	24.13077
98	37.49	35.27846
99	38.08	42.71133
100	40.83	42.71133
101	18.51	20.37700
102	26.36	24.13077
103	20.08	20.37700
104	43.71	42.71133
105	31.61	35.27846
106	28.98	29.37200
107	18.62	20.37700
108	18.64	15.10857
109	13.70	15.10857
110	14.88	15.10857
111	16.46	20.37700
112	11.21	15.10857
113	11.21	15.10857
114	14.18	15.10857
115	20.84	24.13077
116	19.00	24.13077
117	18.07	20.37700

The graph of the observed and predicted values is given below.



Here is the relevant R code.

```
> plot(bodyfat$DEXfat, MB3, pch = 16, col = "red," xlab = "Observed Fat,"
ylab = "Predicted Fat," main = "Regression Tree Output on the bodyfat data")
> abline(a = 0, b = 1, col = "blue")
```

A regression tree can be regarded as non-parametric. It is simple to create and interpret. We do not have to worry about normality and homoscedasticity assumptions. As an exploratory tool, regression tree is a valuable asset. One can work out a random forest of regression trees in the same way we have done for classification tree. This is not pursued here.

11. Boosting using the R package, ada

The purpose of this section is to explain how R can be used for simulations. We want to create a data set artificially with one binary response variable and ten quantitative covariates. Two of the covariates should have a direct bearing on the binary response variable. The data should consist of 500 observations. This is a classification problem. We want to implement the classification tree methodology as the core method in the Boosting protocol. The R commands set out here reflect the way we want to generate the data. For more examples of this nature, look up the R package “GAMBoost” and [Everitt and Hothorn \(2011\)](#).

We want the sample size to be 500.

```
> n <- 500
```

We want to have 10 covariates.

```
> p <- 10
```

The samples are to be generated from a uniform distribution and then manipulate the observations as per the following function:

```
> f <- function(x, a, b, d) a * (x - b)^2 + d
```

*This is just a function. There won't be any output. We will use this function to generate some data. The function has four arguments. The arguments (x, a, b, d) are crunched into a single number $a * (x - b)^2 + d$.*

```
> set.seed(100)
```

Select 250 observations randomly from a uniform distribution on [0, 4].

```
> x1 <- runif(n/2, 0, 4)
```

```
> head(x1)
```

```
[1] 1.2310644 1.0306900 2.2092897 0.2255326 1.8741971 1.9350829
```

For each observation in the data folder x1 apply f with a = -1, b = 2, and d = 1.7 and then add a random observation from the uniform distribution on [-1, 1].

```
> y1 <- f(x1, -1, 2, 1.7) + runif(n/2, -1, 1)
```

```
> head(y1)
```

```
[1] 0.7126735 1.6736598 1.4358352 -1.7051570 2.3704586 2.3902558
```

Draw a random sample of 250 observations from a uniform distribution on [2, 6].

```
> x2 <- runif(n/2, 2, 6)
```

```
> head(x2)
```

```
[1] 2.880930 4.972638 3.223829 3.608835 3.574286 3.763026
```

For each observation in x2, apply the function f with $a = 1$, $b = 4$, and $d = -1.7$ and then add a random observation from a uniform distribution on [-1, 1].

```
> y2 <- f(x2, 1, 4, -1.7) + runif(n/2, -1, 1)
```

Create a column vector with 500 entries with the first 250 being equal to 1 and the remaining equal to 2. This is my response variable.

```
> y <- c(rep(1, n/2), rep(2, n/2))
```

```
> head(y)
```

```
[1] 1 1 1 1 1 1      Draw a random sample of size 500 * 8 from a standard
normal distribution and arrange them in eight columns.
```

```
> mat <- matrix(rnorm(n * 8), ncol = 8)
```

```
> dim(mat)
```

```
[1] 500 8
```

We now create our data. Concatenate x1 and x2. Concatenate y1 and y2. These two are two covariates. The eight columns from "mat" are the other eight covariates.

```
> dat <- data.frame(y = y, x1 = c(x1, x2), x2 = c(y1, y2), mat)
```

```
> dim(dat)
```

```
[1] 500 11
```

```
> head(dat)
```

	x1	x2	x1	x2	x3	x4
1	1.2310644	0.7126735	-1.89839605	3.0922499	0.4723091	0.7470107
2	1.0306900	1.6736598	0.44587930	1.0017852	-0.9579764	-1.0783559
3	2.2092897	1.4358352	0.45626086	-0.4657395	-1.1068111	-0.3425927
4	0.2255326	-1.7051570	-1.52619721	0.5007481	-1.0969130	0.5137962
5	1.8741971	2.3704586	-0.08058262	-1.1339497	0.2825072	1.2232527
6	1.9350829	2.3902558	-0.34001061	-1.2971982	-1.3242908	-0.1001883

	x5	x6	x7	x8
1	0.3366582	-0.07847721	2.05650414	0.8741651
2	1.0867102	1.85286252	0.48994639	-0.8454302
3	0.1397585	1.26858007	0.94617750	0.4267931
4	-0.3429478	-0.42867911	-0.02269179	-0.6505593
5	0.2616749	-0.85582046	1.45329873	-0.5377903
6	0.4183368	0.40065386	0.27509521	1.2460982

There is some confusion with the names of the columns. Change them.

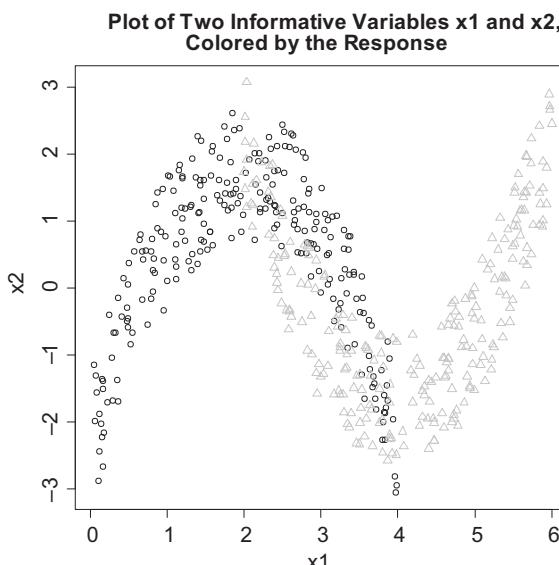
```
> names(dat) <- c("y," paste("x," 1:10, sep = ""))
> head(dat)
```

	y	x1	x2	x3	x4	x5	x6
1	1	1.2310644	0.7126735	-1.89839605	3.0922499	0.4723091	0.7470107
2	1	1.0306900	1.6736598	0.44587930	1.0017852	-0.9579764	-1.0783559
3	1	2.2092897	1.4358352	0.45626086	-0.4657395	-1.1068111	-0.3425927
4	1	0.2255326	-1.7051570	-1.52619721	0.5007481	-1.0969130	0.5137962
5	1	1.8741971	2.3704586	-0.08058262	-1.1339497	0.2825072	1.2232527
6	1	1.9350829	2.3902558	-0.34001061	-1.2971982	-1.3242908	-0.1001883
	x7	x8	x9	x10			
1	0.3366582	-0.07847721	2.05650414	0.8741651			
2	1.0867102	1.85286252	0.48994639	-0.8454302			
3	0.1397585	1.26858007	0.94617750	0.4267931			
4	-0.3429478	-0.42867911	-0.02269179	-0.6505593			
5	0.2616749	-0.85582046	1.45329873	-0.5377903			
6	0.4183368	0.40065386	0.27509521	1.2460982			

Let us reflect on the covariates x_1 and x_2 . The top parts of x_1 and x_2 come from a manipulation of data from the same uniform distribution. The bottom parts come from a manipulation of data from a different uniform distribution. Consequently, x_1 and x_2 should be informative on the makeup of the response variable y . The last eight columns of the data are sheer chaos.

Plot x_1 versus x_2 with reference to the reference variable.

```
> plot(dat$x1, dat$x2, pch = c(1:2)[y], col = c(1,8)[y], xlab = names(dat)[2],
ylab = names(dat)[3], main = "Plot of Two Informative Variables x1 and x2,
Colored by the Response")
```



Now that we have data ready, we will apply Boosting algorithm. We select randomly 100 data points from our given data set and use this as our training set. The rest of the data can be used for validation.

Download the package “ada” and then activate it.

We will use rpart procedure as the underlying classification protocol.

```
> default <- rpart.control()
```

We select 100 data points randomly for Boosting. The sampling is without replacement (replacement = FALSE).

```
> indtrain <- sample(1:n, 100, FALSE)
```

Create the training set.

```
> train <- dat[indtrain,]
```

Create the test set.

```
> test <- dat[-indtrain,]
```

Invoke the ada command. All the covariates x_1 through x_{10} are used.

```
> MB <- ada(y ~ ., data = train, iter = 50, loss = "e," type = "discrete," + control = default)
```

Look at the output.

```
> MB
```

Call:

```
ada(y ~ ., data = train, iter = 50, loss = "e," type = "discrete," control = default)
```

Loss: exponential Method: discrete Iteration: 50

Final confusion matrix for data:

Final prediction

True value	1	2
1	52	0
2	7	41

Train error: 0.07

Out-of-bag error: 0.14 iteration = 10

Additional Estimates of number of iterations:

```
train.err1          train.kap1
```

```
48                  48
```

```
> table(train$y)
```

```
1  2
```

```
52 48
```

The misclassification rate is only 7%. On 110 iterations, the misclassification rate was 14%.

The classification protocol can be applied to the test data set.

```
> MB1 <- addtest(MB, test[, -1], test[, 1])
```

```
> MB1
```

Call:

```
ada(y ~ ., data = train, iter = 50, loss = "e", type = "discrete", control = default)
```

Loss: exponential Method: discrete Iteration: 50

Final confusion matrix for data:

Final prediction

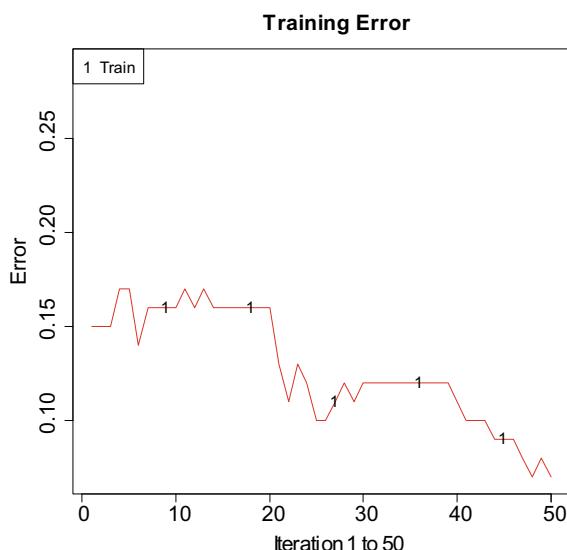
True value	1	2
1	52	0
2	7	41

Train error: 0.07

Out-of-bag error: 0.14 iteration = 10

Additional estimates of number of iterations:

train.err1	train.kap1	test,errs2	test.kaps2
48	48	22	22



The graph gives the history of the misclassification rate with respect to the number of iterations.

12. Epilog

The main emphasis of this article is to present some salient features of the data mining tools bagging and Boosting in classification and regression problems. The underlying classification and regression methodology is explained with sufficient detail using some examples. Random forests building upon classification trees have been enunciated. There is an extensive literature on applications and theoretical underpinnings of the tools presented in the paper. For further information, see Breiman (1992, 1998, 1999a,b, 2001) and Freund and Schapire (1997).

References

- Breiman, Leo., 1992. The little bootstrap and other methods for dimensionality selection in regression: x-fixed prediction error. *J. Am. Stat. Assoc.* 87, 738–754.
- Breiman, Leo, 1996a. Heuristics of instability in model selection. *Ann. Stat.* 24, 2350–2383.
- Breiman, Leo, 1996b. Bagging predictors. *Mach. Learn.* 24, 123–140.
- Breiman, Leo, 1998. Arcing classifiers. *Ann. Stat.* 26, 801–824.
- Breiman, Leo, 1999a. Prediction games and arcng algorithms. *Neural Comput.* 11, 1493–1517.
- Breiman, Leo, 1999b. Combining predictors. In: Sharkey, Amanda (Ed.), *Combining Artificial Neural Nets*. Springer, New York, pp. 31–50.
- Breiman, Leo, 2001. Random forests. *Mach. Learn.* 45, 5–32.
- Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA.
- Everitt, B., Hothorn, T., 2011. *An Introduction to Applied Multivariate Analysis with R*. Springer, New York.
- Freund, Y., Schapire, R., 1996. Experiments with a new Boosting Algorithm. In: *Machine Learning: Proceedings of the 13th International Conference*, pp. 148–156.
- Freund, Y., Schapire, R., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* 55, 1219–1239.
- Zhang, H., Singer, B., 2010. *Recursive Partitioning in the Health Sciences*, second ed. Springer, New York.

This page is intentionally left blank

Matching Score Fusion Methods

Sergey Tulyakov and Venu Govindaraju

*Center for Unified Biometrics and Sensors (CUBS), University at Buffalo,
USA*

Abstract

The matching system can be defined as a type of classifier which calculates the confidence score for each class separately from other classes. Biometric systems are one example of the matching systems. In this chapter we discuss the score fusion methods which are suitable for such systems. In particular, we describe the complexity types of combination methods and characterize some of the existing fusion methods using these types. The higher complexity combination methods account for particular score dependencies typically present in matching systems. We analyze such dependencies and provide suggestions on how more powerful higher complexity combinations can be constructed. The known properties of combination methods are summarized in the five claims, and the theoretical proofs of two claims are provided.

Keywords: classifier combination, matching systems, biometric systems, verification, identification

1. Introduction

It is not unusual for the researchers working on machine learning problem to find not only one, but two or more solutions with acceptable performance. Instead of choosing only one of them and discarding the others, it might be beneficial to somehow combine them and obtain a solution with superior performance. The reasoning for this strategy is that in the combined method the errors of an individual solution will be mitigated by the correctness of other solutions. This reasoning is usually justified by the differences in individual solutions originating from different features, employed methods or training sets, and the combination does indeed succeed in most of the cases.

One of the well-understood combination problems is the problem of combining pattern classifiers. The classifiers C are defined as the methods relating some input patterns I to the set of N classes: $C(I) = \{s_1, \dots, s_N\}$. s_i can have binary

(classification decision), positive integer (rank), or real-value (matching score) type (Xu et al., 1992). Note that if the classifier output is of the first or second types, they can be converted to real values if needed (Lee, 1995). Without loss of generality, we consider the classifiers having matching score outputs. Thus, each score s_i reflects the real-valued confidence of the classifier of recognizing the input I as being from a class i . We will assume that the higher matching score means the higher confidence. In this case, a decision to assign a particular class to the input will consist in finding the maximum of the matching scores: $c = \text{argmax}_i s_i$.

1.1. Performance of classifiers and their combination

Suppose we have a set of K classifiers; classifier j assigns a matching score s_i^j to class i . The set of matching scores $\{s_i^j\}_{i=1,\dots,N, j=1,\dots,K}$ is the input to the combination algorithm. The combined system has the same goal as the component classifiers, therefore its output has the same type. The combination algorithm thus can be defined as a classifier operating on the set of matching scores and having as an output a set of combined scores corresponding to N classes:

$$f(C^1(I), \dots, C^K(I)) = f\left(\left\{s_i^j\right\}_{i=1,\dots,N, j=1,\dots,K}\right) = \{S_1, \dots, S_N\}. \quad (1)$$

In general, different optimization criteria could be used to train the component classifiers, and the combination algorithm could have the same or some other optimization criterion. For example, we might be interested in minimizing the probability of misclassification:

$$\min P(T(I) \neq n | n = \text{argmax}_i S_i), \quad (2)$$

where $T(I)$ denotes the truth class of input I . As another example, we might need to minimize the total cost of misclassification:

$$\min \sum_i C_{n_1 n_2} P(T(I) = n_1) P(n_2 = \text{argmax}_i S_i | T(I) = n_1), \quad (3)$$

where terms $C_{n_1 n_2}$ represent predetermined costs of misclassification of the sample of truth class n_1 as belonging to class n_2 . As a third example, we can allow to make a rejection decision if no class appears to be a clear winner during classification, and introduce additional costs for making such decision; such assumption might come along with the assumption that input I might not belong to any of the defined classes $1, \dots, N$. We can also give an example of fusion problem in a system assuming that some modality might be compromised with some prior probability (Rodrigues et al., 2009); the construction of combination method requires special optimization here.

As illustrated by above examples, we will assume that the performance of classifier combination algorithm is determined by some criterion and is calculated after imposing classification boundaries on the space of combined scores $\{S_1, \dots, S_N\}$. The classification regions can correspond to one of N classes or the class of “rejects.” The preimages of these regions on the score space of the original classifiers, $\{s_i^j\}_{i=1,\dots,N, j=1,\dots,K}$, are defined as the classification regions of the combination algorithm.

One of the questions frequently asked by the researchers is whether the combination of classifiers always performs better than the single classifier, or the combination of a smaller subset of classifiers. The following claim addresses this question assuming the existence of optimal combination algorithm, that is the algorithm delivering a classification region corresponding to the minimum of the defined optimization criterion.

Claim 1 *The performance of the optimally combined set of classifiers is no worse than the performance of any individual classifier or optimally combined subset of classifiers in that set.*

Indeed, one of the possible combination functions in Eq. (1) could have an output coinciding with the output of a single classifier: $f(C^1(I), \dots, C^K(I)) = C^j(I)$. Optimal combination function, by its definition, should have no worse performance than any other combination function, and in particular, no worse performance than above combination function with the output of a single classifier. Similarly, if $f_L(C^{k_1}(I), \dots, C^{k_L}(I))$ is an optimal combination function defined on a subset of classifiers, then $f(C^1(I), \dots, C^K(I)) = f_L(C^{k_1}(I), \dots, C^{k_L}(I))$ will be one of the combination functions defined on a whole set, and its performance will be no better than the performance of the optimal combination function defined on a whole set of classifiers.

According to the above claim, by using additional classifier and properly combining it with previously used classifiers, we can achieve the performance improvement and, in the worst case, to have the same performance as before. But the improvement in performance is not guaranteed. For example, consider two classifiers with binary decision outputs and suppose that the second classifier makes correct classification decision only when the first classifier does. In this case, the addition of the second classifier does not improve performance—the optimal combination algorithm will simply output the decisions of the first classifier.

Although the situation where an additional classifier has nothing to contribute to already existing set of combined classifiers is possible, we speculate that it is not a main reason why the combination method sometimes will not improve its performance. Many publications in the area of classifier combination have reports that one or the other method has worse performance than some component classifier (e.g., Kittler et al., 1998; Alkoot and Kittler, 1999; Snelick et al., 2005). It seems that using the limited set of combination functions, which do not approximate the ideal optimal combination function, or using improper optimization criteria is a more frequent reason for the failure. In the Appendix A we provide an example of the poor optimization criteria resulting in a combination failure. Another reason for failure, discussed in the next section, concerns the ability of the learning algorithms for approximations given limited number of training samples.

1.2. Statistical learning for classifier combinations

In the settings of Bayesian classification theory, the optimal classification decision can be achieved based on the knowledge of the distributions of feature vectors and the prior probabilities of the classes. In respect to classifier combination problem, we need to know the densities of the scores $p(\{s_i^j\}_{i=1,\dots,N, j=1,\dots,K} | T(I) = n)$ and the

prior class probabilities $P(T(I) = n)$. Let us consider optimization criterion given in Eq. (2). The optimal classifier should assign the class which maximizes posterior class probability

$$\operatorname{argmax}_n P\left(T(I) = n \mid \left\{s_i^j\right\}_{i=1, \dots, N, j=1, \dots, K}\right). \quad (4)$$

Therefore, if we take the combined score to be a posterior class probability

$$\begin{aligned} S_n &= f_n\left(\left\{s_i^j\right\}_{i=1, \dots, N, j=1, \dots, K}\right) = P\left(T(I) = n \mid \left\{s_i^j\right\}_{i=1, \dots, N, j=1, \dots, K}\right) \\ &= \frac{p\left(\left\{s_i^j\right\}_{i=1, \dots, N, j=1, \dots, K} \mid T(I)=n\right)P(T(I)=n)}{\sum_{n=1, \dots, N} p\left(\left\{s_i^j\right\}_{i=1, \dots, N, j=1, \dots, K} \mid T(I)=n\right)P(T(I)=n)}, \end{aligned} \quad (5)$$

this will give us optimal solution to the combination problem with optimization criterion of Eq. (2).

Ordinarily, the only information which is available to the combination algorithm is the set of matching scores $\{s_i^j\}_{i=1, \dots, N, j=1, \dots, K}$ of combined classifiers. The combination problem can be stated as a problem of constructing secondary classifier operating in this score space. It can be solved by trying to approximate score densities and using optimal combination algorithm given by above equations. Alternatively, a variety of generic classification methods developed in the pattern classification field can be used to solve the problem. The following question arises from these considerations: is the classifier combination problem different from the problem of constructing secondary classifier? More specifically, does the knowledge about the nature of the scores $\{s_i^j\}$ help us to construct better combination solutions than the solutions simply accepting scores $\{s_i^j\}$ as a feature vector?

The answer to these questions seems to be positive, and the classifier combination problem can be defined as a problem of constructing a simpler secondary classifier operating on a set of matching scores, which still includes an optimal decision in its search space but requires a smaller number of training samples than a generic classifier. In this chapter we will discuss the particular ways in which this construction can be achieved.

1.3. Classifier ensembles

Throughout this chapter we will use the assumption that in order to learn the combination method we have sufficient training data. This data is generally different from the data which is used to train the classifiers. By having this assumption we separate our task from the task of combining ensemble classifiers (Rokach, 2010). In the ensemble classifier framework a single training set is typically used to generate classifiers and to train the combination function. The specific behavior of individual classifiers in the ensemble is hard to measure and the combination methods usually rely only on some general performance measure, such as the overall classification rate on the training dataset.

One accepted explanation of the improvements observed in classifier ensembles methods is based on the decomposition of the added error of the classifiers into bias and variance components (Kong and Dietterich, 1995; Tibshirani, 1996; Breiman, 1996). Tumer and Ghosh (1999) (see also Kuncheva, 2002; Ludmila, 2004; Fumera and Roli, 2002, 2004) associate trained classifiers with the approximated feature vector densities of each class. In this framework, trained classifiers provide approximations to the true posterior class probabilities or to the true class densities:

$$f_i^m(x) = p_i(x) + \epsilon_i^m(x),$$

where i is the class index and m is the index of trained classifier. For a fixed point x the error term can be represented as a random variable where randomness is determined by the random choice of the classifier or used training set. By representing it as a sum of mean β (bias of the trained classifier) and zero-mean random variable η (variance of the trained classifier) we get

$$\epsilon_i^m(x) = \beta_i(x) + \eta_i^m(x).$$

For simplicity, assume that the considered classifiers are unbiased, that is $\beta_i(x) = 0$ for any x, i . If point x is located on the decision boundary between classes i and j then the added error of the classifier is proportional to the sum of the variances of η_i and η_j :

$$E_{\text{add}}^m \sim \sigma_{\eta_i^m}^2 + \sigma_{\eta_j^m}^2.$$

If we average M such trained classifiers and if error random variables η_i^m are independent and identically distributed as η_i , then we would expect the added error to be reduced M times:

$$E_{\text{add}}^{\text{ave}} \sim \sigma_{\eta_i^{\text{ave}}}^2 + \sigma_{\eta_j^{\text{ave}}}^2 = \frac{\sigma_{\eta_i}^2 + \sigma_{\eta_j}^2}{M}.$$

The application of the described theory is very limited in practice since a large number of assumptions about classifiers are required (Kuncheva, 2004). Besides independence assumption of errors, the error distributions, that is the distributions of the random variable η_i , have to be modeled. Note that η_i is the difference between true distribution $p_i(x)$ and our best guess about this distribution. If we knew what the difference is, we would have been able to improve our guess in the first place. Some research works (Kuncheva, 2002; Alkoot and Kittler, 1999) try to make some assumptions about these estimation error distributions and see which combination rule is better for a particular hypothesized distribution.

Due to inability to fully learn the behavior of classifiers, only simple combination methods like weighted sum are possible for training the ensemble combination function. In contrast, in our work we assume that the separate training set is sufficient for learning such individual characteristics of classifiers as the densities of the score vectors $\{s_1^j, \dots, s_N^j\}$. Then, more complex combination methods are possible; we will explore and categorize them in this chapter.

2. Matching systems

Among the classifiers with matching score outputs we can distinguish a particular type, which we will call *matchers*, or *matching systems*. For the classifiers of such type, the matching score of each class can be derived separately from the matching scores of other classes. We can represent the matcher in a function form $M(I, k) = s_k$ (“for image I and class k we produce a matching score s_k ”), and the N -class classifier associated with matcher could be written as $C(I) = \{M(I, 1), \dots, M(I, N)\} = \{s_1, \dots, s_N\}$. In contrast to traditionally defined N -class classifiers, the number of classes in matching systems could be variable, and the addition or removal of some classes from consideration will not influence the matching score assigned to a particular class.

Many of the traditional pattern classifiers might not be matchers. For example, k -nearest neighbor classifier has a matching score for class i as a proportion of training samples of class i in k -neighborhood relative to all samples. If we drop all other classes from consideration, then the k -neighborhood would contain only samples of class i , and the classifier’s score will always be 1. As another example, a fully connected neural network trained to classify an image into one of 10 classes of digits and having 10 output nodes (Lee, 1995) is not a matcher; if we wanted to classify both 26 latin letters and 10 digits, we would have to change the configuration and retrain it. In this case, the presence of other classes will influence the output results and the scores of modified neural network would most certainly be different.

At the same time, some similar approaches to classification can be considered as matching systems. If we use k -nearest neighbor method to approximate the densities of each class (using the volume occupied by k -nearest samples of each class (Theodoridis and Koutroumbas, 1999)) and output the values of the densities as matching scores, then such system will be a matching system. In the example of neural networks, if for each of 10 digits we train a neural network separating images of that digit from the images of all other digits or other symbols, then the matching score for each class will be calculated by its own network and we will obtain a matching system as a result.

One of the ways to construct matching systems is to keep the data about each class separate from the data related to other classes. Such data related to a single class is called a *template* of the class and can be denoted as T_k . The system could also have a single class-independent function calculating the correspondences between stored class templates and current input: $M(I, k) = f(I, T_k)$. The goal of the system designer is to construct the best representation of the class data in a class template, T_k , and to implement the matching function f . The classes could be added to such system by the *enrollment* procedure, i.e., by extracting and storing their corresponding templates.

2.1. Biometric matching systems

The biometric systems are typically constructed as matching systems of above-described type. Each person constitutes one’s own class, and the number of classes in the biometric system equals to the number of enrolled persons. Since this number

is typically large and new persons can be added at any time, it makes little sense to try to learn traditional N -class classifier. Another obstacle to N -class learning is the frequent lack of data—it would be unreasonable to ask the person to supply multiple biometric scans during enrollment so that a sufficient number of instances of a class will be available during classifier's training. In contrast, the biometric templates of the person usually consist of the vector of measurements taken from a single biometric scan, and the matching function can calculate a matching score as a distance between two such vectors—one enrolled and one obtained from an input pattern to be classified.

Although there is a big diversity in the biometric modalities, biometric template structures, and matching functions, practically all biometric systems follow the above definition of matching systems. Among the simpler approaches to template creation and matching, we can name hand geometry, iris, and face recognition, where a fixed length feature vector of measurements can be extracted and the Euclidean distance between such vectors serves as matching function. The examples of more complex biometric templates include fingerprint templates consisting of the variable size sets of minutia points, signature and gait templates having a variable length sets of measurement vector. For such templates, the matching function might search for some transformation producing a correspondence between two templates and the likelihood of such transformation will represent the matching score.

In biometric systems, a *genuine* score is produced when two templates of the same person (or same class) are matched, and an *impostor* score is produced while matching different persons (or different classes). The system is typically run in *verification mode*, where the person is matched against a single enrolled template and a decision of having genuine or impostor match is made, or *identification mode*, where the person is matched against all enrolled templates and the best match is chosen (this mode corresponds to traditional classification decisions). We will use these terms in this chapter when we speak about matching systems.

2.2. Handwritten word recognizers

As another example of the matching systems we can consider the handwritten word recognition systems ([Kim and Govindaraju, 1997](#)). In such systems, the set of possible classes, the word lexicon, can be given in advance, and the task of the word recognizer is to tell which of the lexicon words corresponds to the input image. As in the case of biometric matchers we might not have enough data to train traditional classification system; the lexicon might be large and include words which are not even present in the training set.

The training of the recognizer in [Kim and Govindaraju \(1997\)](#) includes the training of character recognizer. Since the lexicon word is represented as a sequence of characters, the word will be recognized if the sequence of subimages is recognized as a sequence of characters of that word. Thus, the template of the lexicon word can be thought of as a set of character templates present in this word. The matching score directly corresponds to the distance between input image and the character-based word model. For each lexicon word this score is calculated separately, and the definition of matching system is satisfied.

3. Selected approaches to fusion in matching systems

By definition, the matching scores for different classes in the matching systems are calculated separately. Intuitively, since matching scores for classes other than class i have no influence on matching scores s_i^j of class i , the combination function of Eq. (1) might calculate the combined score for each class separately:

$$S_i = f(s_i^1, \dots, s_i^K). \quad (6)$$

As we discuss in Section 5, the set of combination functions given by this equation is not equivalent to the set of combination functions of Eq. (1). Nevertheless, the combination methods of Eq. (6) greatly simplify the learning of the combination function and most published research in the area of matching system fusion seeks for the combination methods represented by this equation.

It is interesting to note, that many combination methods of traditional classifiers did not have the simplifying formulation of Eq. (6), and have been developed assuming the full complexity of Eq. (1): neural network combinations (Lee, 1995), Dempster–Shafer theory-based fusion (Xu et al., 1992), Behavior-Knowledge Spaces (Huang and Suen, Huang 1995), rank-based combination methods (Ho et al., 1994). In contrast, the combination methods for matching systems seem to begin with simplified assumptions of Eq. (6) and introduce more complexity as additional parameters to this formula.

In this section we will review some existing methods of score combinations in matching systems. Since the goal of this chapter is to describe the methods of constructing proper complex combination methods of Eq. (1), we are more interested in the combination methods which go beyond simplified methods of Eq. (6).

3.1. Fusion rules and score normalization

Fusion rules could be defined as a simple implementation of Eq. (6) using some operator for score combination. For example, the sum rule is defined as a simple summation of scores: ($S_i = s_i^1 + \dots + s_i^K$), product rule as product, and so on. Kittler et al. (1998) justified the use of some of these rules by making different assumptions on the nature of combined scores s_i^j . As example, the assumptions might include the independence of scores produced by different matchers, the assumption that scores approximate posterior class probabilities, the assumption that the approximation errors originating from limited training data have particular distributions, and so on. It is hard to expect that the scores from a general matching system will satisfy any of the described assumptions, and, as a consequence, the optimality of the combination algorithm is not guaranteed.

The application of fusion rules necessitates the use of score normalization and a variety of score normalization methods have been proposed for use in biometric matching systems. The experimental evaluations seem to favor different methods of normalization and subsequent application of fusion rules (Jain et al., 2005; Snelick et al., 2005). It is also not clear what normalization method and what fusion rule could approximate the optimal Bayesian combination method. For example, the tanh normalization followed by sum fusion reported to have a superior performance

in Jain et al. (2005) is a parametric method with only few learnable parameters and could hardly approximate arbitrary boundaries of Bayesian method.

The important question which we addressed in our work (Tulyakov and Govindaraju, 2006) is whether the use of normalization followed by the application of optimal combination rule has any advantage over directly learning optimal combination function of Eq. (6). We assumed that two biometric matchers of different modalities are available to us and tried to implement the optimal Bayesian fusion method in two different ways. The first way consisted in approximating two-dimensional score densities, and the second way consisted in approximating one-dimensional score densities of each matcher and multiplying them (using independence property of matchers of different modalities) to obtain required two-dimensional densities. For a particular approximation method (Parzen windows) we showed both theoretically and experimentally that approximation of one-dimensional densities and their subsequent multiplication has better performance than direct approximation of two-dimensional score densities. As a result of that research, we can stipulate the following claim:

Claim 2 *The combination algorithm learning the behavior of each classifier separately (e.g., by learning score normalization functions) and optimally combining individually normalized score can achieve better performance than the combination algorithm attempting learning in the common score space.*

The consequence of the claim is that, indeed, by performing proper score normalization and applying optimal fusion rule we can achieve better results than by directly learning combination function of Eq. (6). The unsolved question remains, what are these proper score normalization method and fusion rule? For a particular case of independent matchers, the proper normalization could be converting a score to the value of class likelihood ratio and the fusion rule could be a product rule. But it is not clear, if the same normalization and fusion rule will remain optimal for dependent matchers. Note also that a number of possible pairs of normalizations and fusions could be optimal—if we take the logarithm of likelihood ratio to be the normalization function, then we should be adding normalized scores instead of multiplying. Also, the experimental results show (Tulyakov and Govindaraju, 2006) that the performance improvement is rather small. Thus, if we do not know proper normalization and fusion functions, then we most probably will be better off with direct approximation of optimal combination function.

3.2. Dynamic classifier selection

The idea behind the dynamic classifier selection (DCS) method (Woods et al., 1997; Giacinto and Roli, 2001) is to find out which of the combined classifiers performs best for a particular input to be classified and taking the output of this classifier as the output of the combination algorithm. In order to find the best classifier, some training samples similar to the input are found, and the performance of each classifier is summarized on these training samples. The similar training samples could be searched in the original feature space of the classifiers (Woods et al., 1997) or in the common score space of combined classifiers (Giacinto and Roli, 2001).

By discerning the strengths of the combined classifiers with the help of training samples (not used when those classifiers were originally trained) the DCS method seems to deliver good performance, and it might be possible to also prove that given sufficient training data it will perform no worse than any component classifier. Note that DCS methods could be of type conforming to Eq. (6) (Giacinto and Roli, 2001), or might use some information extraneous to it, like location of templates in original feature space (Woods et al., 1997).

DCS methods might not deliver the optimal classifier combination performance. For example, suppose we have N -class classification problem with large N , say 1000. Suppose we have two classifiers outputting ranks of the classes and that the correct class has ranks from 1 to 5 with equal probability in any classification trial for both classifiers. The other classes are getting their ranks assigned randomly. If the DCS method performs well, we can expect that it will select the classifier if this classifier has correct class on top. One classifier has correct class on top with probability .2, and for two classifiers the correct class will be on top in at least one of them with probability $1 - (1 - .2)(1 - .2) = .36$. On the other hand, if a Borda count combination method is used (consisting in the addition of class ranks), it will give correct decision with the probability near 1.0.

The above example suggests that though DCS methods have desirable characteristic of better performance than component classifiers, it can be easily outperformed by other combination methods, and hardly delivers optimal performance. Essentially, the information from non-selected classifiers is getting discarded in the process, resulting in inferior performance.

3.3. User-and template-specific combinations

Some of the matcher combination methods try to introduce more complex combination decisions than those defined in Eq. (6). The straightforward improvement of that formula would consider the combination functions to be specific for each class:

$$S_i = f_i(s_i^1, \dots, s_i^K). \quad (7)$$

Jain and Ross (2002) derived user-specific thresholds in verification problem, as well as, used different weights for different users in the weighted sum combination function. Fierrez-Aguilar et al. (2005) performed user-specific estimation of the score densities utilized for the calculation of fused score. Toh et al. (2004) separated the task of providing user-specific decisions and user-specific fused score calculation, and argued that employing both techniques at the same time achieves superior results.

One of the problems underscored in this research is the difficulty of reliably approximating the user-specific genuine score densities. It might be difficult to obtain the multiple samples belonging to the same user in the considered biometric applications. The same problem stands when the attempts to perform class-specific score normalization for handwritten word recognizers were made (Bouchaffra et al., 1999; Ianakiev and Govindaraju, 2002).

Another possible approach to expanding Eq. (6) involves the consideration of scores assigned to classes other than i while calculating the combined score S_i for class i . Different types of background and cohort models have been considered

in the area of speech recognition (Rosenberg and Parthasarathy, 1996; Colombi et al., 1997), as well as, score normalizations which involve such score sets, e.g., T-normalizations (Auckenthaler et al., 2000; Sturim and Reynolds, 2005). The approaches considering the ranks of the scores (Brunelli and Falavigna, 1995) implicitly use the sets of scores assigned to different classes as well, and could be considered of this type.

The approaches presented above follow the goal of research in classifier combinations—constructing learnable algorithms approximating the optimal combination method of Eq. (1). In Section 5 we describe in more detail the relationships among these approaches. And in Section 6 we provide a technique of constructing the combination methods of this type by utilizing statistics of score sets.

3.4. Utilizing external knowledge

In few situations, some knowledge about the properties of the combined matchers is available, and this knowledge might not be reflected in the matching scores. It could be possible to incorporate such knowledge into the combination algorithm and thus significantly improve its performance. For example, the fusion method of Tulyakov and Govindaraju (2006) utilizes the knowledge of the statistical independence of matching scores produced by different matchers. This knowledge is not directly accessible from the training matching scores. Certainly, we can apply some statistical criteria and prove that independence hypothesis is true with some probability, but in the case if hypothesis is not true we might end up with worse performance than before. As another example, a reference implementation of NIST fingerprint matcher includes functions for measuring quality of fingerprint images, and the measures of quality could be used alongside the matching scores for making authentication decisions or for their normalization (Grother and Tabassi, 2007).

A number of fusion methods have been presented recently utilizing some measures of the quality of the templates or the data which was used to create them. Poh and Kittler (2012) provide an extensive list of references to such works. Generally, better quality of the template used by a particular matcher implies more confident matching results obtained by this matcher and should lead to bigger weights given to it. But instead of such heuristic considerations, the quality measures could be directly incorporated into the learnable combination function. The quality measures derived separately from the template's features are probably most effective, but it might be possible to also derive implicit quality measures from the sets of matching scores (Kumar and Zhang, 2010).

A different type of external information used during fusion could originate not from the quality of templates but from some parameters of the matching algorithm. For example, during the fusion of face and gait biometric matchers we might want to check if both matchers correctly estimated the pose of the person or any other environment conditions (Geng et al., 2010). If both matchers agree on the context, then we are more certain in their results. In another example from the area of handwritten word recognition, the closeness of lexicon word to other words is estimated (Govindaraju et al., 2002). Such estimate provides a measure of difficulty in recognizing a particular lexicon word and can be used to optimize the decision thresholds of the recognizer.

Judging by the success of reports utilizing external knowledge during fusion, we can state the following claim:

Claim 3 *Incorporating external knowledge not reflected in the combined scores can typically improve the performance of the combination method.*

4. Operating modes of matching systems

In this section we will assume a simplified approach to fusion represented by the Eq. (6). Matching systems operating in verification mode separate two classes: genuine and impostor verification attempts. By considering the combination task as a pattern classification problem in the K -dimensional space, Bayesian minimization of the misclassification cost results in the likelihood ratio combination function (Tulyakov et al., 2007):

$$f_{lr}(s^1, \dots, s^K) = \frac{p_{\text{gen}}(s^1, \dots, s^K)}{p_{\text{imp}}(s^1, \dots, s^K)}, \quad (8)$$

where p_{gen} and p_{imp} are K -dimensional densities of score tuples $\{s^1, \dots, s^M\}$ corresponding to the two classes—genuine and impostor verification attempts. For a particular fusion problem, we can estimate the densities p_{gen} and p_{imp} from the training data and use the above formula to calculate the combined score and threshold it.

Now, suppose our system operates in an identification mode and we want to check whether the likelihood ratio function found to be optimal for verification mode will be also optimal for identification mode. Suppose we performed a match of the input sample by all K matchers against all N classes and obtained KN matching scores $\{s_i^j\}_{i=1, \dots, N; j=1, \dots, K}$. Assuming equal prior class probabilities, the Bayes decision theory states that in order to minimize the misclassification rate, the sample should be classified as the one with the highest value of the likelihood function $p(\{s_i^j\}_{i=1, \dots, N; j=1, \dots, K} | \omega_i)$. Thus, for any two classes ω_1 and ω_2 we can classify the input as ω_1 rather than ω_2 if

$$p(\{s_i^j\}_{i=1, \dots, N; j=1, \dots, K} | \omega_1) > p(\{s_i^j\}_{i=1, \dots, N; j=1, \dots, K} | \omega_2). \quad (9)$$

Let us make an assumption that the scores assigned to each class are sampled independently from scores assigned to other classes; scores assigned to genuine class are sampled from the K -dimensional genuine score density, and scores assigned to impostor classes are sampled from the K -dimensional impostor score density:

$$\begin{aligned} p\left(\left\{s_i^j\right\}_{i=1, \dots, N; j=1, \dots, K} | \omega_i\right) \\ = p\left(\left\{s_1^1, \dots, s_1^K\right\}, \dots, \left\{s_{\omega_i}^1, \dots, s_{\omega_i}^K\right\}, \dots, \left\{s_N^1, \dots, s_N^K\right\} | \omega_i\right) \\ = p_{\text{imp}}(s_1^1, \dots, s_1^K) \dots p_{\text{gen}}(s_{\omega_i}^1, \dots, s_{\omega_i}^K) \dots p_{\text{imp}}(s_N^1, \dots, s_N^K). \end{aligned} \quad (10)$$

After substituting (10) in (9) and canceling out the common factors, we obtain the following inequality for accepting class ω_1 (rather than ω_2):

$$\begin{aligned} p_{\text{gen}}(s_{\omega_1}^1, \dots, s_{\omega_1}^K) p_{\text{imp}}(s_{\omega_2}^1, \dots, s_{\omega_2}^K) \\ > p_{\text{imp}}(s_{\omega_1}^1, \dots, s_{\omega_1}^K) p_{\text{gen}}(s_{\omega_2}^1, \dots, s_{\omega_2}^K) \end{aligned} \quad (11)$$

or

$$\frac{p_{\text{gen}}(s_{\omega_1}^1, \dots, s_{\omega_1}^K)}{p_{\text{imp}}(s_{\omega_1}^1, \dots, s_{\omega_1}^K)} > \frac{p_{\text{gen}}(s_{\omega_2}^1, \dots, s_{\omega_2}^K)}{p_{\text{imp}}(s_{\omega_2}^1, \dots, s_{\omega_2}^K)}. \quad (12)$$

The terms in each part of the above inequality are exactly the values of the likelihood ratio function f_{lr} calculated for classes ω_1 and ω_2 . Thus, the class maximizing the KN -dimensional likelihood function of inequality (9) is the same as the class maximizing the K -dimensional likelihood ratio function of inequality (12). Thus the likelihood ratio combination rule is optimal under the assumption of score independence.

In Tulyakov et al. (2007, 2010) we provide the examples of real-life matchers showing that the assumption of score independence does not hold. Moreover, in one case the likelihood ratio combination method has worse performance than the single matcher. It turns out the following claim holds.

Claim 4 *There exist matching systems in which optimal combination functions for verification and identification modes are different.*

We give the proof for this claim in Appendix A. The example, provided during the proof, can have parameters for which the likelihood ratio combination method performs worse than a single matcher, and thus simulates the observed behavior of real-life matcher with the same property.

It seems to be difficult to derive the analytic form of the optimal combination function for systems operating in identification mode. In Tulyakov et al. (2007) we presented two heuristic optimization methods for identification mode modeling the likelihood ratio function for genuine and some specially chosen impostor scores. And Tulyakov and Govindaraju (2009) explore the methods of optimizing the neural network combination method for best performance in identification mode. Although the good experimental results were presented in these works, the proof of the optimality of constructed methods is still missing.

5. Complexity types of classifier combination methods

As we mentioned in Section 1.2, the goal of the classifier combination research is to provide a simpler way of combining classifiers than by learning combination function of Eq. (1). At the same time, as we pointed in Section 3, the simple combination rules of Eq. (6) might not provide the sufficient discriminating ability similar to the one of Eq. (1). Therefore, it would be desirable to find some intermediate methods possessing the positive qualities of both equations. One possible categorization of such methods is provided in this section.

According to [Tulyakov and Govindaraju \(2006\)](#), we can define the following four categories of combination methods:

1. Low complexity methods: $S_i = f(\{s_i^j\}_{j=1,\dots,K})$. Methods of this type require only one combination function to be trained, and the combination function takes as input scores for one particular class as parameters.
2. Medium complexity I methods: $S_i = f_i(\{s_i^j\}_{j=1,\dots,K})$. Methods of this type have separate score combining functions for each class and each such function takes, as input parameters, only the scores related to its class.
3. Medium complexity II methods: $S_i = f(\{s_i^j\}_{j=1,\dots,K}, \{s_n^j\}_{j=1,\dots,K; n=1,\dots,N, n \neq i})$. Methods of this type take as parameters not only the scores related to the same class, but all output scores of classifiers. Combination scores for each class are calculated using the same function, but scores for class i are given a special place in the parameter list. Applying function f for different classes effectively means permutation of the function's parameters.
4. High complexity methods: $S_i = f_i(\{s_n^j\}_{j=1,\dots,K; n=1,\dots,N})$. Functions calculating final scores are different for all classes, and they take as parameters all the scores output by the base classifiers.

We already gave some examples of the combination methods of low and medium I complexity types, as they are represented by Eqs. (6) and (7) correspondingly. Rank-based combination methods and methods employing T-normalization can be considered as belonging to medium II type, since the whole set of matching scores related to all enrolled classes is taken into consideration during fusion. The high complexity combinations mostly appear in the fusion of traditional classifiers with small number of classes.

Figure 1 illustrates the relationships between presented complexity types of combinations. Medium complexity types are subsets of high complexity combinations, and the set of low complexity combinations is exactly the intersection of sets of medium I and medium II combination types. In order to avoid a confusion in terminology we will define a combination method to belong to a particular type only if it belongs to this type and does not belong to the more specific type.

In [Tulyakov \(2006\)](#) we provided a stricter description of these complexity types using the concept of VC (Vapnik–Chervonenkis) dimension ([Vapnik, 1998](#)). In particular, we derived the formulas for VC dimensions for each complexity type, and showed how complexity can be reduced either by adopting a lower complexity type combination or by restricting the set of trainable combination functions. The ability

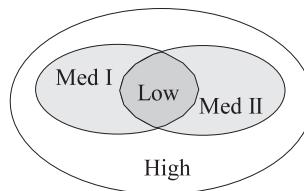


Fig. 1. The relationship diagram of different combination complexity types.

to use VC dimension for characterization of different combination types justifies our usage of term “complexity types.”

The proof that the lower complexity combination types constitute proper subsets in corresponding higher complexity combination methods is formulated in the next claim.

Claim 5 *There exist combination problems in which the methods of higher complexity outperform the best methods of lower complexity types.*

In Appendix B we provide the proof that the low complexity type is the proper subset of medium II complexity type. The proof that low complexity type is the proper subset of medium I complexity type could follow from it if we notice the duality between enrolled and test templates (Tulyakov et al., 2008). Whereas the medium II complexity combinations try to model the dependence of matching scores between a single test template and a set of enrolled templates, medium I complexity combinations can be constructed by modeling the dependence of matching scores between a single enrolled template and a set of test templates. Due to symmetrical nature of these models, we can deduce the required property of Claim 5.

6. Modeling matching score dependencies

In order to construct effective combination methods of higher complexity types, it is important to understand which information is contained in the additional sets of scores which are getting discarded in low complexity types. Let us compare, for example, the formulas for combination of low and medium II complexity type. Assuming a single matcher (or one matcher in a combination of few independent matchers), we have corresponding formulas $f(s_i)$ and $f(\{s_i\}, \{s_n\}_{n=1, \dots, N, n \neq i})$. If there is some type of dependence between score s_i and the set of scores $\{s_n\}_{n=1, \dots, N, n \neq i}$, then the particular parameters of this trial could be estimated from scores $\{s_n\}_{n=1, \dots, N, n \neq i}$ and, consequently, we could judge the expected behavior of s_i in this matching trial. Effectively, we perform a comparison of s_i against all other scores $\{s_n\}_{n=1, \dots, N, n \neq i}$, and more precisely estimate the probability of s_i to be genuine or impostor.

Clearly, it is not necessary to keep all the scores $\{s_n\}_{n=1, \dots, N, n \neq i}$ to obtain the properties of the current matching attempt. Instead, we could derive a small set of parameters, or statistics of the set $\{s_n\}_{n=1, \dots, N, n \neq i}$. Thus, instead of combination formula $f(\{s_i\}, \{s_n\}_{n=1, \dots, N, n \neq i})$, we can consider simplified formula $f(\{s_i\}, \{t_l\}_{l=1, \dots, L})$, with $L \ll N$. In Tulyakov and Govindaraju (2008) we present a more detailed derivation of score set statistics-based combination algorithm.

The question on what statistics should be extracted from the score sets is probably the most difficult one in the research of matching system fusion. The scores are produced by the matching algorithm, and there might exist some peculiarities in score distributions which are hard to model. The papers from speech recognition community report usually good performance of T-normalization method which is defined by two parameters—mean and standard deviation of a score set. In our research, we usually get better performance from utilizing order statistics, such as second best score (Tulyakov et al., 2008). Few of the statistics could be used at the same time for better modeling as well.

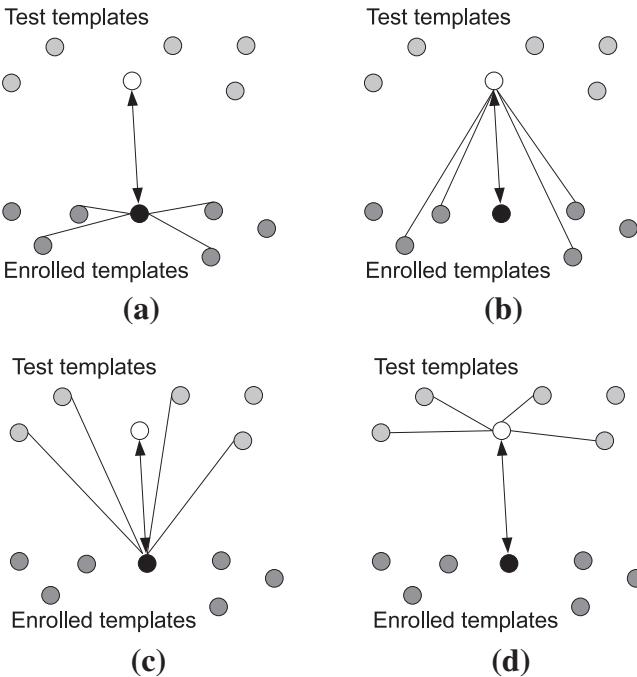


Fig. 2. Different scenarios for constructing score dependence models. The connector with arrow represents the one score which is currently considered by the combination methods and needs to be thresholded or normalized. Other connectors represent the set of reference matching scores.

The sets of scores considered above in the formula for medium II complexity combinations are not the only one choice. In Fig. 2 we present possible architectures and related score sets, which can be used along with one particular score. The score sets could consist of mostly impostor scores (which we have in abundance in typical matching systems), or could specifically designed to include some genuine scores.

Not all architectures of Fig. 2 are equally utilized in practice. The architecture of Fig. 2a is probably most intuitive and extensively used in cohort methods. For this architecture it is possible to find the cohort set of templates before the matching. The distances to cohort templates give us a measure of the difficulty of the classification problem. Similar measures can also be used in dynamic classifier selection methods (Section 3.2). The architecture of Fig. 2b is implemented by rank-based methods and by the methods using T-normalization. It requires the calculation of matching score related to a set of enrolled classes during the matching of a single test input. The architecture of Fig. 2c can be implemented by keeping track on how a set of test templates was matched to a single enrolled template. Our paper (Tulyakov et al., 2008) describes implementation of such method. Finally, the architecture of Fig. 2d seems to be difficult to implement and would require keeping the set of test templates for future matching.

Note that all sets of scores in presented architectures are derived by matching a single template; the dependence of scores comes from this fact. Architectures

of Fig. 2a and c have the score sets derived using single enrolled template. Therefore, the score sets reflect the property of this template, and the methods using this architecture will naturally create enrolled template specific combination functions. The term “user specific” in the published literature usually refers to this type of combinations. Analogously, architectures of Fig. 2b and d result in test template-specific combinations. The “T” in T-normalizations originates from word “test” and thus correctly reflects its purpose. In terms of the complexity types, the enrolled template-specific combinations can typically be represented as medium I complexity combinations, and test template-specific combinations can be represented as medium II complexity combinations.

Note that a combination method might involve more than one score dependence architecture. In Cheng et al. (2011b) we presented the results of utilizing the dependences of both architectures of Fig. 2b and c. The constructed combination method is of high complexity—it belongs to both medium I and medium II complexity types. As expected, it outperforms both methods of medium complexity types, which, in turn, have better performance than low complexity combination.

7. Score combination applications

7.1. Verification and identification systems

The approach described in the previous section provides a complete way of constructing possibly high complexity combinations in matching systems. First, for each combined matcher we have to select the architectures of scores dependences of Fig. 2 which will be most practical and at the same time potentially deliver the best performance improvements. Second, the dependences in the score sets should be modeled by selecting proper statistics of score sets. Third, the statistics should be used alongside the single scores in combination formulas.

As an example, let us consider the Eq. (5) giving the solution to classification problem with the criteria of minimizing misclassification error. It requires approximation of NK -dimensional densities $p(\{s_i^j\}_{i=1,\dots,N, j=1,\dots,K} | \omega_n)$. Since N can be large in matching systems, we have to reduce the dimensionality by replacing score sets with some statistics. First we separate the score of class under interest from the scores of other classes:

$$p\left(\left\{s_i^j\right\}_{i=1,\dots,N, j=1,\dots,K} | \omega_n\right) \sim p\left(\left\{s_n^j, \left\{s_i^j\right\}_{i=1,\dots,N, i \neq n}\right\}_{j=1,\dots,K} | \omega_n\right)$$

and then replace score sets by statistics:

$$p\left(\left\{s_n^j, \left\{s_i^j\right\}_{i=1,\dots,N, i \neq n}\right\}_{j=1,\dots,K} | \omega_n\right) \sim p\left(\left\{s_n^j, \left\{t_l^j\right\}_{l=1,\dots,L}\right\}_{j=1,\dots,K} | \omega_n\right).$$

The last term represents the density in $(L + 1)K$ -dimensional space and can be more easily approximated. If the scores from different matchers are independent, then additional decomposition and better approximation (according to Claim 2) are possible. The density training in the verification problems can be further simplified by

“forgetting” the identity of class ω_n ; as we discussed in Section 4 such a simplification might not work for identification problems.

7.2. Performance prediction

One interesting application of the considered methods of modeling score dependencies is the prediction of identification system performance. Suppose, we have some results of testing the matching system in identification mode for a small number of enrolled persons, e.g., 100. Is it possible to say what the performance of such system will be if the number of enrolled persons is increased to, say, 10,000?

In Tulyakov and Govindaraju (2011) we addressed this problem in detail. The main idea was to simulate the run of the system with larger number of persons, and for each simulated identification attempt we needed to sample the impostor scores according to the correct distribution. It turns out that it is possible to properly sample scores, if we group together the scores from available identification trials. The grouped trials should have similar characteristics and the closeness of trials could be defined by the closeness of specially chosen score set statistics. The statistics should fully reflect the distribution of scores in the trial, and, therefore, the problem of choosing the best performing statistics is equivalent to the problem of choosing best statistics for fusion of matchers.

The experimental results confirmed that the order statistics was usually better than other statistics. But in three out of four considered matchers, the mean and standard deviation statistics of T-normalization also worked well.

7.3. Multiple samples

Suppose the matching system needs to verify the person’s identity and obtains the person’s biometric scan. After matching it to the enrolled template, the results do not have enough confidence to ascertain person’s identity. The system might ask the person to provide a second biometric scan of the same modality, and try to match it to enrolled template again. A second matching score is now available and decision can be made by using it. But the first matching score is available to us as well, and instead of using only single second score, we can try to combine both score for more confident decision. Such fusion of matching scores derived with the help of single enrolled template and repeated test template is called *multi-sample* fusion. We can also consider an alternative architecture with multiple enrolled samples of one person, and a single or multiple test scans.

The general combination approach utilizing dependences in score sets of Section 6 works in this situation as well. The score sets and score set statistics can be extracted for each individual sample, and the combination algorithm could incorporate the statistics of different samples in the same way as statistics related to templates of different matchers were used. Cheng et al. (2011a) presents exactly this algorithm for combining scores in multisample fusion problem.

A different approach was considered in Cheng et al. (2012). The main observation of this paper is that if the presented samples are too similar, it makes little sense to combine their matching results, and the combined score should be reduced. But, if samples are different, they carry more information and bigger confidence should

be given to the combination made using them. The difference between samples can be derived from the sets of matching scores. But using score set statistics might not work well in this case. Instead, a direct comparison of score vectors was performed, and the distance between score vectors were used as a measure of sample similarity. This measure was given to the combination function as a parameter, and the training confirmed that bigger difference between samples led to increased combined matching score.

8. Conclusion

In this chapter we gave the overview of the score fusion methods in matching systems. First, we exactly defined the problem of classifier combination as a problem of constructing a combination function of scores having lesser complexity than the generic secondary classifier operating in a score space of component classifiers. Second, we defined the matching systems as a subset of classifiers having scores derived separately from other classes. Third, we defined verification and identification modes of operation for matching systems, and proved that optimal combination methods might be different for them. Fourth, we defined the complexity types of score combination functions, and proved their difference. Fifth, the methods of creating combination methods of different complexity types by utilizing score set statistics were introduced.

Many of existing score fusion methods for matching systems can be described in terms of framework used in the chapter. We gave examples of fusion rules, score normalization techniques, dynamic classifier selection, user-specific and test template-specific fusion methods, and the methods using some information external to the combined matching scores. Furthermore, we reviewed additional tasks, performance prediction, and multi-sample fusion, which can be solved with considered methods.

Appendices

A. Proof of Claim 4

In order to prove the Claim 4, we have to construct an example of matching system which has different optimal combination functions for verification and identification operating modes. We will set the number of matchers to be 2, and the number of classes, N , equal to 2. Thus matcher $j, j = 0$ or 1, outputs two scores s_1^j and s_2^j , with one of these scores being genuine, s_{gen}^j , and the other score being impostor, s_{imp}^j . Suppose, that the scores of matchers are sampled from bivariate normal distribution: $\{s_{\text{gen}}^j, s_{\text{imp}}^j\} \sim N(\{1, 0\}, \Sigma_j)$, with

$$\Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \text{ and } \Sigma_2 = \begin{pmatrix} 1 & \lambda \\ \lambda & 1 \end{pmatrix}.$$

Thus the two scores of matcher 1 are independent and the two scores of matcher 2 are dependent if $\lambda \neq 0$. The marginal distributions of genuine scores of both matchers are normal $N(1,1)$ and the marginal distributions of impostor scores of both matchers are normal $N(0,1)$. Further, we will assume that the scores related to two matchers are independent; so the joint distribution of two genuine scores is normal $\{s_{\text{gen}}^1, s_{\text{gen}}^2\} \sim N(\{1,1\}, I)$ and the joint distribution of two impostor scores is normal $\{s_{\text{imp}}^1, s_{\text{imp}}^2\} \sim N(\{0,0\}, I)$, I is unit matrix.

If this system operates in verification mode, then the optimal score combination function, likelihood ratio, has same contours as $s^1 + s^2 = c$, and therefore we can take $f_{lr}(s^1, s^2) = s^1 + s^2$ as our optimal combination function for verification system. The distributions of genuine, $\{s_{\text{gen}}^1, s_{\text{gen}}^2\}$, and impostor, $\{s_{\text{imp}}^1, s_{\text{imp}}^2\}$, score pairs do not depend on λ , and optimal combination function for verification system, $f_{lr}(s^1, s^2)$, is the same for any choice of λ . But, as we show next, for identification system the situation is different: optimal combination function and the performance of the combined system will depend on λ .

Now we will derive the optimal combination function for identification system in our example. Suppose that in one identification trial we obtained the following scores from both matchers: $\{s_1^1, s_2^1\}$ from matcher 1 and $\{s_1^2, s_2^2\}$ from matcher 2. The combination function f produces combined scores $S_1 = f(s_1^1, s_1^2)$ and $S_2 = f(s_2^1, s_2^2)$. By comparing S_1 and S_2 we decide which of two classes, 1 or 2, is genuine or impostor. In order to minimize the classification error, we have to use optimal Bayesian classification: classify the sample as class 1 instead of class 2 if and only if

$$\begin{aligned} & p\left(\{s_1^1, s_2^1\}, \{s_1^2, s_2^2\} \mid \text{class 1 is genuine}\right) \\ & > p\left(\{s_1^1, s_2^1\}, \{s_1^2, s_2^2\} \mid \text{class 2 is genuine}\right). \end{aligned} \quad (13)$$

So, the optimal combination function f should be such that $f(s_1^1, s_1^2) > f(s_2^1, s_2^2)$ if and only if Eq. (13) holds. After utilizing the independence of matchers ($p(\{s_1^1, s_2^1\}, \{s_1^2, s_2^2\} | \dots) = p(\{s_1^1, s_2^1\} | \dots)p(\{s_1^2, s_2^2\} | \dots)$), we substitute the given normal densities of score pairs produced by each matcher:

$$p\left(\{s_1^j, s_2^j\} \mid \text{class 1 is genuine}\right) = N\left(\begin{pmatrix} s_1^j \\ s_2^j \end{pmatrix}; \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \Sigma_j\right), \quad (14)$$

$$p\left(\{s_1^j, s_2^j\} \mid \text{class 2 is genuine}\right) = N\left(\begin{pmatrix} s_2^j \\ s_1^j \end{pmatrix}; \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \Sigma_j\right). \quad (15)$$

After substitution, we can transform Eq. (13) into the following inequality:

$$s_1^1 + s_1^2 * \frac{1}{1 - \lambda} > s_2^1 + s_2^2 * \frac{1}{1 - \lambda}. \quad (16)$$

Therefore, we can take the following function as the optimal combination function for identification system in our example:

$$f_{id}(s^1, s^2) = s^1 + s^2 * \frac{1}{1 - \lambda}. \quad (17)$$

Table 1

Identification system performance (the frequency of top score being genuine) of single matchers and their combinations in the proof of Claim 4

λ	0	.3	.5	.7	-.5
Matcher 1	76.01%	76.01%	76.01%	76.01%	76.01%
Matcher 2	76.01%	80.08%	84.11%	90.13%	71.81%
Likelihood Ratio	84.13%	86.09%	87.58%	89.25%	81.44%
Optimal Combination	$s_1 + s_2$	$s_1 + \frac{10}{7}s_2$	$s_1 + 2s_2$	$s_1 + \frac{10}{3}s_2$	$s_1 + \frac{2}{3}s_2$
Optimal Performance	84.13%	86.47%	88.96%	92.95%	81.93%

Table 1 presents some performance numbers of single matchers derived by numerically integrating joint density of scores $p(s_{\text{gen}}^2, s_{\text{imp}}^2) = N(\{1, 0\}, \Sigma_2)$ over the area $s_{\text{gen}}^2 > s_{\text{imp}}^2$. The identification system performance increases with the increase of λ . Intuitively this can be explained as following: if we have a positive correlation between genuine and impostor scores, for a high impostor score we have bigger probability that genuine will also be high, and the identification attempt will still succeed; similarly for low genuine scores we have bigger probability of even lower impostor scores. For negatively correlated scores ($\lambda < 0$) we observe a decrease in performance.

In order to calculate the identification system performance of combination function f we numerically integrate $p(s_{\text{gen}}^1, s_{\text{gen}}^2, s_{\text{imp}}^1, s_{\text{imp}}^2) = p(s_{\text{gen}}^1, s_{\text{imp}}^1)p(s_{\text{gen}}^2, s_{\text{imp}}^2)$ over the region $f(s_{\text{gen}}^1, s_{\text{gen}}^2) > f(s_{\text{imp}}^1, s_{\text{imp}}^2)$. The performance of the likelihood ratio combination function $f = f_{lr}$ is given in the fourth row of Table 1. Its performance reflects the change in performance of matcher 2: the better matcher 2 performs, the better is the performance of likelihood ratio combination. But notice that for large values of λ (e.g., $\lambda = .7$) the performance of likelihood ratio is getting worse than the performance of the single matcher 2. The decrease in performance clearly indicates that likelihood ratio might not be an optimal combination function for identification systems.

Combination based on f_{id} coincides with the combination based on f_{lr} only when $\lambda = 0$. In other cases, f_{id} performs better than f_{lr} in identification operating mode. The last two rows of Table 1 contain samples of optimal combination function f_{id} for identification mode and corresponding correct identification rates. In all cases f_{id} performs better than any single matching participating in combination.

B. Proof of Claim 5

In this section we prove that the set of medium II complexity combinations is different from the set of low complexity combinations. Though this fact can be intuitively obvious, we present a strict mathematical proof by constructing examples of identification systems in which optimal low complexity combination performs worse than optimal medium II complexity combination. As we discussed in Section 4 the optimal combination algorithms are different for verification and identification modes of system operation if there is a dependence between matching scores assigned to different classes. Thus we construct two examples, one for verification and another

for identification operating modes, to prove the difference in complexity types for both operating modes.

For verification mode of operation we consider a system with only one matcher and with two enrolled classes. Suppose that genuine and impostor scores are distributed according to normal distributions: $s_{\text{gen}} \sim N(1,1)$ and $s_{\text{imp}} \sim N(0,1)$. Also supposed that in every verification trial, if we calculated the matching scores for both enrolled classes, we would always get: $s_{\text{imp}} = s_{\text{gen}} - 1$. Thus, if this system is run in identification mode, we will always correctly place genuine score on top. Let s_1 and s_2 be two matching scores assigned to two classes, one genuine and another impostor. The low complexity combination has to rely on only a single score of a claimed class. Suppose a class 1 is claimed and the only decision we can make is by comparing score s_1 to some threshold θ : accept verification attempt if $s_1 > \theta$. Clearly, there could be verification trials where class 1 is impostor, but its score is bigger than the threshold and it is accepted, or trials where class 1 is genuine, but it is rejected since its score is less than the threshold. So, $\text{FRR}(\theta) > 0$ and $\text{FAR}(\theta) > 0$ for any θ . On the other hand, we can consider the following decision algorithm of medium II complexity type: accept claimed identity i if $s_i > s_{3-i}$, and reject otherwise. Such algorithm will have $\text{FRR} = \text{FAR} = 0$. So we proved that for verification mode of operation the optimal low complexity combination is not able to achieve same performance as optimal medium II complexity combination.

For identification mode of operation we consider a system with two matchers and two classes. Let these matchers be independent. Let the matching scores be only 0 or 1 and let the probabilities of score outputs for first (s_1^1, s_2^1) and second (s_1^2, s_2^2) matchers, in case first class (ω_1) is a genuine class, be given in Table 2. Let also assume that if second class (ω_2) is genuine, then corresponding score pairs probabilities are the same (with permutation): $P(s_1^1, s_2^1 | \omega_2 \text{ is genuine}) = P(s_2^1, s_1^1 | \omega_1 \text{ is genuine})$ and $P(s_1^2, s_2^2 | \omega_2 \text{ is genuine}) = P(s_2^2, s_1^2 | \omega_1 \text{ is genuine})$.

Table 2
Probabilities of matching score outputs for classifiers
in identification operating mode example

s_1^1	s_2^1	$P(s_1^1, s_2^1 \omega_1 \text{ is genuine})$
0	0	0.1
0	1	0.1
1	0	0.4
1	1	0.4

s_1^2	s_2^2	$P(s_1^2, s_2^2 \omega_1 \text{ is genuine})$
0	0	0.1
0	1	0.2
1	0	0.5
1	1	0.2

Table 3
Optimal low complexity combination function for identification operating mode example

s_i^1	s_i^2	$f(s_i^1, s_i^2)$
0	0	0
0	1	2
1	0	1
1	1	3

Low complexity combination for such system operating in identification mode will be represented by the formula:

$$C = \arg \max_{i=1,2} f(s_i^1, s_i^2)$$

and since scores s_i^j are only of two values (0 and 1) we can enumerate all possible combination functions as having values {0,1,2,3} on different pairs {0,0},{0,1}, {1,0},{1,1}. After considering all such possible combination functions, we find that function f shown in Table 3 gives the best correct identification rate of 0.62.

Note that due to the symmetry in original score distributions $P(s_1^1, s_2^1 | \omega_1$ is genuine) for different genuine classes ω_1 and ω_2 , we cannot have class-specific combinations. Thus the set of low complexity combinations coincides with the set of medium I combinations, and the set of medium II combinations coincides with the set of high complexity combinations. So, the optimal combination of medium II complexity type is exactly the optimal combination of high complexity type, which is the optimal classification algorithm on the set of all scores $(s_1^1, s_1^2, s_2^1, s_2^2)$. Since we know distributions of scores are assigned by our matchers, using Bayes theorem and matcher independence assumption ($P(s_1^1, s_2^1, s_1^2, s_2^2 | \omega_1 \text{ is genuine}) = P(s_1^1, s_2^1 | \omega_1 \text{ is genuine})P(s_1^2, s_2^2 | \omega_1 \text{ is genuine})$), we can calculate posterior probabilities of both classes for each combination of score outputs ($P(\omega_i \text{ is genuine} | s_1^1, s_2^1, s_1^2, s_2^2)$) and perform optimal Bayesian classification. Such classification achieves correct identification rate of .65. Thus, the medium II complexity combination achieves better performance in identification operating mode than optimal low complexity combination. Note that in both cases we had the same undecided rate of .15, where the value of combination function is the same for both classes or the posterior class probability is the same for both classes.

References

- Alkoot, F.M., Kittler, J., 1999. Experimental evaluation of expert fusion strategies. Pattern Recogn. Lett. 20 (11–13), 1361–1369.
- Auckenthaler, Roland, Carey, Michael, Lloyd-Thomas, Harvey, 2000. Score normalization for text-independent speaker verification systems. Digit. Signal Process. 10 (1–3), 42–54.
- Bouchaffra, D., Govindaraju, V., Srihari, S., 1999. A methodology for mapping scores to probabilities. IEEE Trans. Pattern Anal. Mach. Intell. 21 (9).
- Breiman, L., 1996. Bagging predictors. Mach. Learn. 24 (2), 123–140.

- Brunelli, R., Falavigna, D., 1995. Person identification using multiple cues. *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (10), 955–966.
- Cheng, Xi, Tulyakov, S., Govindaraju, V., 2011a. Combination of multiple samples utilizing identification model in biometric systems. In: International Joint Conference on Biometrics (IJCB2011), Washington, USA.
- Cheng, Xi, Sergey, Tulyakov, Venu, Govindaraju, 2011b. Combination of user- and enrollee-specific statistical information in verification systems. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2011, pp. 126–131.
- Cheng, Xi, Tulyakov, S., Govindaraju, V., 2012. Utilization of matching score vector similarity measures in biometric systems. In: Computer Vision and Pattern Recognition Workshops, Providence, USA.
- Colombi, J.M., Reider, J.S., Campbell, J.P., 1997. Allowing good impostors to test. In: Thirty-First Asilomar Conference on Signals, Systems and Computers, 1997, vol. 1, pp. 296–300.
- Fierrez-Aguilar, Julian, Garcia-Romero, Daniel, Ortega-Garcia, Javier, Gonzalez-Rodriguez, Joaquin, 2005. Bayesian adaptation for user-dependent multimodal biometric authentication. *Pattern Recogn.* 38 (8), 1317–1319.
- Fumera, Giorgio, Roli, Fabio, 2002. Performance analysis and comparison of linear combiners for classifier fusion. In: SSPR/SPR, pp. 424–432.
- Fumera, Giorgio, Roli, Fabio, 2004. Analysis of error-reject trade-off in linearly combined multiple classifiers. *Pattern Recogn.* 37 (6), 1245–1265.
- Geng, Xin, Smith-Miles, Kate, Wang, Liang, Li, Ming, Wu, Qiang, 2010. Context-aware fusion: a case study on fusion of gait and face for human identification in video. *Pattern Recogn.* 43 (10), 3660–3673.
- Giacinto, Giorgio, Roli, Fabio, 2001. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recogn.* 34 (9), 1879–1881.
- Govindaraju, V., Slavik, P., Hanhong Xue, 2002. Use of lexicon density in evaluating word recognizers. *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (6), 789–800.
- Grother, P., Tabassi, E., 2007. Performance of biometric quality measures. *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (4), 531–543.
- Ho, T.K., Hull, J.J., Srihari, S.N., 1994. Decision combination in multiple classifier systems. *IEEE Trans. Pattern Anal. Mach. Intell.* 16 (1), 66–75.
- Huang, Y.S., Suen, C.Y., 1995. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (1), 90–94.
- Ianakiev, K., Govindaraju, V., 2002. Deriving pseudo-probabilities of correctness given scores. In: Chen D., Cheng, X. (Eds.), *Pattern Recognition and String Matching*. Kluwer Publishers.
- Jain, A.K., Ross, A., 2002. Learning user-specific parameters in a multibiometric system. In: International Conference on Image Processing, 2002, vol. 1, pp. I-57–I-60.
- Jain, Anil, Nandakumar, Karthik, Ross, Arun, 2005. Score normalization in multimodal biometric systems. *Pattern Recogn.* 38 (12), 2270–2285.
- Kim, Gyeonghwan, Govindaraju, V., 1997. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Trans. Pattern Anal. Mach. Intell.* 19 (4), 366–379.
- Kittler, J., Hatef, M., Duin, R.P.W., Matas, J., 1998. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.* 20 (3), 226–239.
- Kong, E., Dietterich, T., 1995. Error-correcting output coding corrects bias and variance. In: 12th International Conference on Machine Learning, pp. 313–321.
- Kumar, A., Zhang, D., 2010. Improving biometric authentication performance from the user quality. *IEEE Trans. Instrum. Meas.* 59 (3), 730–735.
- Kuncheva, L.I., 2002. A theoretical study on six classifier fusion strategies. *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (2), 281–286.
- Lee, Dar-Shyang, 1995. Theory of classifier combination: the neural network approach. Ph.D Thesis, SUNY at Buffalo.
- Kuncheva, Ludmila I., 2004. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley InterScience.
- Poh, N., Kittler, J., 2012. A unified framework for biometric expert fusion incorporating quality measures. *IEEE Trans. Pattern Anal. Mach. Intell.* 34 (1), 3–18.
- Rodrigues, Ricardo N., Ling, Lee Luan, Govindaraju, Venu, 2009. Robustness of multimodal biometric fusion methods against spoof attacks. *J. Vis. Lang. Comput.* 20 (3), 169–179.
- Rokach, Lior, 2010. Ensemble-based classifiers. *Artif. Intell. Rev.* 33 (1), 1–39.

- Rosenberg, A.E., Parthasarathy, S., 1996. Speaker background models for connected digit password speaker verification. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 1996, ICASSP-96, vol. 1, pp. 81–84.
- Snelick, R., Uludag, U., Mink, A., Indovina, M., Jain, A., 2005. Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems. *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (3), 450–455.
- Sturim, D.E., Reynolds, D.A., 2005. Speaker adaptive cohort selection for tnorm in text-independent speaker verification. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005 (ICASSP '05), vol. 1, pp. 741–744.
- Theodoridis, S., Koutroumbas, K., 1999. *Pattern Recognition*. Academic Press.
- Tibshirani, R., 1996. Bias, variance and prediction error for classification rules. Technical Report 41, Department of Statistics, University of Toronto.
- Toh, K.-A., Jiang, Xudong, Yau, Wei-Yun, 2004. Exploiting global and local decisions for multimodal biometrics verification. *IEEE Trans. Signal Process.* 52 (10), 3059–3072 (see also *IEEE Transactions on Acoustics, Speech, and Signal Processing*).
- Tulyakov, S., 2006. A complexity framework for combination of classifiers in verification and identification systems. PhD Thesis, State University of New York at Buffalo.
- Tulyakov, S., Govindaraju, V., 2006. Classifier combination types for biometric applications. In: Conference on Computer Vision and Pattern Recognition Workshop, 2006, CVPRW '06, p. 58.
- Tulyakov, Sergey, Govindaraju, Venu, 2006. Utilizing independence of multimodal biometric matchers. In: International Workshop on Multimedia Content Representation, Classification and Security, Istanbul, Turkey.
- Tulyakov, S., Govindaraju, V., 2008. Use of identification trial statistics for combination of biometric matchers. *IEEE Trans. Inform. Forensics Security* 3 (4), 719–733.
- Tulyakov, Sergey, Govindaraju, Venu, 2009. Neural network optimization for combinations in identification systems. In: Benediktsson, Jon Atli, Kittler, Josef, Roli, Fabio (Eds.), Eighth International Workshop on Multiple Classifier Systems, MCS 2009. of Lecture Notes in Computer Science, vol. 5519. Springer, Reykjavik, Iceland, pp. 418–427.
- Tulyakov, Sergey, Govindaraju, Venu, 2011. Predicting performance in large-scale identification systems by score resampling. In: Bhangu Bir, Govindaraju Venu (Eds.), *Multibiometrics for Human Identification*. Cambridge University Press, pp. 363–388 (Cambridge Books Online).
- Tulyakov, Sergey, Govindaraju, Venu, Wu, Chaohong, 2007. Optimal classifier combination rules for verification and identification systems. In: Seventh International Workshop on Multiple Classifier Systems, Prague, Czech Republic.
- Tulyakov, Sergey, Wu, Chaohong, Govindaraju, Venu, 2007. Iterative methods for searching optimal classifier combination function. In: First IEEE International Conference on Biometrics: Theory, Applications, and Systems, BTAS 2007, pp. 1–5.
- Tulyakov, Sergey, Li, Jiang, Govindaraju, Venu, 2008. Enrolled template specific decisions and combinations in verification systems. In: IEEE Second International Conference on Biometrics: Theory, Applications and Systems (BTAS 08).
- Tulyakov, Sergey, Zhang, Zhi, Govindaraju, Venu, 2008. Comparison of combination methods utilizing t-normalization and second best score model. In: CVPR 2008 Workshop on Biometrics.
- Tulyakov, S., Wu, C., Govindaraju, V., 2010. On the difference between optimal combination functions for verification and identification systems. *Int. J. Pattern Recogn. Artif. Intell.* 24 (2), 173–191.
- Tumer, K., Ghosh, J., 1999. Linear and order statistics combiners for pattern classification. In: Sharkey, A.J.C. (Ed.), *Combining Artificial Neural Nets: Ensembles and Multi-Net Systems*. Springer-Verlag, London, pp. 127–162.
- Vapnik, V.N., 1998. *Statistical Learning Theory*. Wiley, New York.
- Woods Jr. K., Kegelmeyer, W.P., Bowyer, K., 1997. Combination of multiple classifiers using local accuracy estimates. *IEEE Trans. Pattern Anal. Mach. Intell.* 19 (4), 405–410.
- Xu, L., Krzyzak, A., Suen, C.Y., 1992. Methods for combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. Syst. Man Cybern.* 23 (3), 418–435.

This page is intentionally left blank

Part II: Object Recognition

This page is intentionally left blank

Statistical Methods on Special Manifolds for Image and Video Understanding

Pavan Turaga¹, Rama Chellappa², and Anuj Srivastava³

¹*School of Arts, Media, Engineering, School of Electrical, Computer, Energy Engineering, Arizona State University, Tempe, AZ, USA*

²*Department of Electrical and Computer Engineering and UMIACS, University of Maryland, College Park, MD 20742, USA*

³*Department of Statistics, Florida State University, Tallahassee, FL, USA*

Abstract

In this chapter we describe some recent advances in image and video understanding applications based on statistical inference methods over nonlinear manifolds. These tools are developed by adapting techniques from multivariate statistics to the differential geometries of the manifold of interest. These manifolds include: (1) transformation groups, such as rotation, translation, and scale, (2) their quotient spaces, and (3) shape manifolds of planar curves. We discuss applications of statistical methods on the aforementioned manifolds for face recognition, shape analysis, activity recognition, age estimation, and gesture recognition. In each case we motivate the need for manifold-based methods and discuss techniques from differential geometry that are needed to design appropriate statistical inference algorithms for these applications. These discussions contain some illustrative examples and references to the literature for additional reading.

Keywords: analytic manifolds, Riemannian shape metrics, alignment, face recognition, activity recognition

1. Introduction

In the field of computer vision, some of the fundamental mathematical models for describing physical aspects such as shape, illumination, and motion have often been described in the language of non-Euclidean geometry. For example, the shape observed in an image from landmarks is often viewed as the information that remains after nuisance factors such as position on the image, rotations, etc. have

been filtered out, and the resulting spaces are often nonlinear domains. Illumination analysis for convex Lambertian objects have resulted in cone models for image appearance (Georghiades et al., 2001). 2D and 3D motion of objects in space is often parameterized in terms of rotation matrices, which are intrinsically non-Euclidean in nature (Kanatani, 1990; Grenander et al., 1998; Moakher, 2002). These fundamental models are often used in the context of inverse problems such as structure from motion, photometric stereo, or filtering problems including motion-tracking. In order to ensure that the solutions obtained adhere to the appropriate mathematical constraints, one needs to understand how to extend familiar optimization and algebraic tools to these structured spaces.

In addition, recent advances in image- and video-based recognition techniques have resulted in a wealth of features or models—such as shape (Kendall, 1984), color/intensity histograms (Hafner et al., 1995), SIFT (Lowe, 2003), histogram of gradients (Dalal and Triggs, 2005), linear dynamic systems (Soatto et al., 2001), covariance matrices (Tuzel et al., 2006), etc. Many of these features and models, however, do not live in a Euclidean space. What this means is that the underlying distance metric on the space is not the usual L_2 norm but a highly non-linear function which is also in most cases hard to compute. As an example, shape spaces have long been considered as Riemannian manifolds—Kendall’s shape space is a complex spherical manifold (Kendall, 1984), affine shape spaces are Grassmann manifolds (Begelfor and Werman, 2006), and the space of closed-curves also forms a manifold (Michor and Mumford, 2006; Younes, 1998; Mio et al., 2007). The space of $d \times d$ symmetric positive definite matrices or tensors forms a cone in the space of matrices, thus is not a vector space (Pennec et al., 2006; Tuzel et al., 2006). The space of linear subspaces is a quotient space of the orthogonal group leading to the Grassmann manifold. Linear dynamic systems can be identified by the column-space of the observability matrix which is also understood as a Grassmann manifold (Turaga et al., 2011). Even the commonly used histograms form a simplex in \mathbb{R}^n and do not live in a vector space. To extend classical algorithms for statistical modeling, classification, regression, etc. to these structured spaces, one needs to take recourse to the Riemannian geometric properties of these spaces. This naturally leads to definitions of distances and statistics in terms of geodesics on these manifolds.

The explicit use of manifolds, Lie groups, and Lie group actions on manifolds in pattern recognition area was pioneered by Ulf Grenander. His formulation of pattern theory for representation of complex systems is based on using combinations of simple elementary units, called *generators*, which under the actions of small groups capture the variability exhibited by the observed systems. This theory was presented in a series of monographs (Grenander, 1976, 1981), culminating in the textbook (Grenander, 1993). In terms of problems in image understanding and computer vision, some applications of Grenander’s pattern theory appeared in the joint works with Amit et al. (1991), Grenander et al. (1990), and Grenander and Miller (1994).

2. Some motivating examples

We briefly enumerate here a few application domains in image and video understanding that require appreciating the geometry of some non-Euclidean manifolds:

- (1) *Rigid motion:* Models for describing motion of objects in the real world often involve descriptions in the form of translations and rotations. The set of all such rigid motions is easily modeled as a product space between the translation vectors and rotation matrices, also known as the special Euclidean group. This forms the basis of many model-based detection and tracking algorithms. Here, one is provided with a 3D model of an object of interest, and the goal is to find the optimal translation and rotation parameter so that the object is detected in the given image. This often takes the form of an optimization problem, where the domain of optimization is the special Euclidean group.
- (2) *Shape analysis:* Many tasks such as object recognition and action recognition involve understanding the shapes of objects or human silhouettes. This typically begins with the extraction of features such as certain landmark points on the boundaries of the objects. In the context of recognition, it is important to achieve invariance to certain nuisance factors such as translation, scale, and rotation. Under these invariance requirements, it can be shown that the shape space is equivalent to a complex spherical manifold. Further, by requiring full affine invariance, it can be shown that shapes reside on a Grassmann manifold. More recently, shapes have been considered to be continuous closed planar curves. The space of such curves can also be characterized as a manifold.
- (3) *Dynamical models:* Dynamical models are an integral component of many video-based algorithms such as gait analysis, action recognition, and video-based face recognition. In these applications, features are extracted from each frame of the video, and the sequence of features is compactly parameterized using appropriate low-dimensional dynamical models. One popular dynamical model for such time-series data is the linear dynamical system (LDS). The space spanned by the columns of the observability matrix of the LDS model can be identified as a point on the Grassmann manifold. This allows recognition to be posed as a classification problem on the Grassmann manifold.

As these examples illustrate, manifolds arise quite naturally in several vision-based applications. In the rest of the chapter, we shall summarize some basic tools on manifolds for learning and inference. We shall then discuss specific examples in image and video understanding to illustrate their practical impact.

3. Differential geometric tools

We start by considering the definition of a general differentiable manifold. A topological space M is called a *differentiable manifold* if, among other properties, it is *locally Euclidean*. This means that for each M , there exists a neighborhood and a mapping that *flattens* that neighborhood. The Euclidean space \mathbb{R}^n is an n -dimensional differentiable manifold which can be covered by a single neighborhood. Any open subset of a differentiable manifold is itself a differentiable manifold. $GL(n)$, the set of all $n \times n$ non-singular matrices, i.e., $GL(n) = \{A \in M(n) | \det(A) \neq 0\}$, is an open subset of $\mathbb{R}^{n \times n}$. Thus, it is also a differentiable manifold.

In order to perform differential calculus, i.e., to compute gradients, directional derivatives, critical points, etc. of functions on manifolds, one needs to understand

the tangent structure of those manifolds. Although there are several ways to define tangent spaces, one intuitive approach is to consider differentiable curves on the manifold passing through the point of interest, and to study the velocity vectors of these curves at that point. The set of all such tangent vectors is called the *tangent space* to M at p . Even though the manifold M maybe nonlinear, the tangent space $T_p(M)$ is always linear and one can impose probability models on it using more traditional approaches. In case of the Euclidean space \mathbb{R}^n , the tangent space $T_p(\mathbb{R}^n) = \mathbb{R}^n$ for all $p \in \mathbb{R}^n$. For any $A \in GL(n)$, let $\gamma(t)$ be a path in $GL(n)$ passing through $A \in GL(n)$ at $t = 0$. The velocity vector at p , $\dot{\gamma}(0)$, is an $n \times n$ matrices. Hence, $T_A(GL(n)) = \mathbb{R}^{n \times n}$.

We now consider the task of measuring distances on a manifold. This is accomplished using a Riemannian metric defined as follows:

Definition 1. A *Riemannian metric* on a differentiable manifold M is a map $\langle \cdot, \cdot \rangle$ that smoothly associates to each point $p \in M$ a symmetric, bilinear, positive definite form on the tangent space $T_p(M)$.

A differentiable manifold with a Riemannian metric on it is called a *Riemannian manifold*. As an example, \mathbb{R}^n is a Riemannian manifold with the Riemannian metric $\langle v_1, v_2 \rangle = v_1^T v_2$, the standard Euclidean product. Similarly, for the unit sphere \mathbb{S}^n and a point $p \in \mathbb{S}^n$, the Euclidean inner product on the tangent vectors make \mathbb{S}^n a Riemannian manifold. That is, for any $v_1, v_2 \in T_p(\mathbb{S}^n)$, we used the Riemannian metric $\langle v_1, v_2 \rangle = v_1^T v_2$. Using the Riemannian structure, it becomes possible to define lengths of paths on a manifold. Let $\alpha : [0, 1] \mapsto M$ be a parameterized path on a Riemannian manifold M that is differentiable everywhere on $[0, 1]$. Then $\frac{d\alpha}{dt}$, the velocity vector at t , is an element of the tangent space $T_{\alpha(t)}(M)$, with length given by $\sqrt{\left\langle \frac{d\alpha}{dt}, \frac{d\alpha}{dt} \right\rangle}$. The length of the path α is then given by

$$L[\alpha] = \int_0^1 \sqrt{\left\langle \left(\frac{d\alpha(t)}{dt}, \frac{d\alpha(t)}{dt} \right) \right\rangle} dt. \quad (1)$$

For any two points $p, q \in M$, one can define the distance between them as the infimum of the lengths of all smooth paths on M which starts at p and ends at q :

$$d(p, q) = \inf_{\{\alpha : [0, 1] \mapsto M | \alpha(0) = p, \alpha(1) = q\}} L[\alpha]. \quad (2)$$

A path $\hat{\alpha}$ which achieves a local minimum of $L[\alpha]$, if it exists, is a *geodesic* between p and q on M . Geodesics on a unit sphere \mathbb{S}^n are great circles. The distance minimizing geodesic between two points p and q is the shortest of the two arcs of a great circle joining between them. As a parameterized curve, this geodesic is given by:

$$\alpha(t) = \frac{1}{\sin(\theta)} [\sin(\theta - t)p + \sin(t)q], \quad (3)$$

where $\theta = \cos^{-1}(\langle p, q \rangle)$.

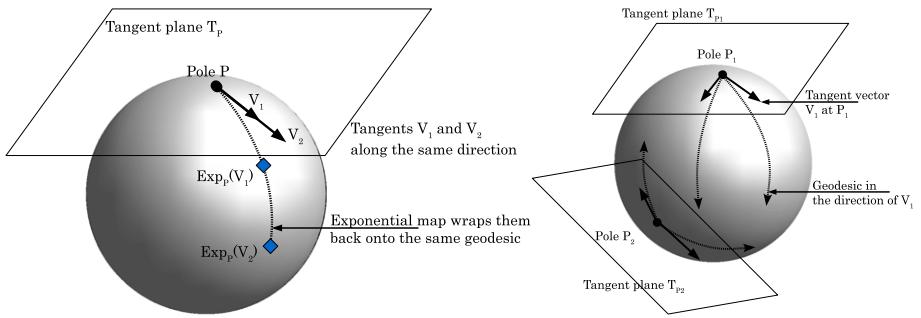


Fig. 1. Illustration of tangent spaces, tangent vectors, and geodesics. P_1 and P_2 are points on the manifold. T_{P_1} and T_{P_2} are the tangent spaces at these points. Note that there is a unique local mapping between the manifold and the tangent plane and this local mapping depends upon the pole. Geodesics paths are constant velocity curves on the manifold. Tangent vectors correspond to velocities of curves on the manifold.

An important tool in studying statistics on a manifold is an exponential map. If M is a Riemannian manifold and $p \in M$, the *exponential map* $\exp_p : T_p(M) \rightarrow M$, is defined by $\exp_p(v) = \alpha_v(1)$ where α_v is a constant speed geodesic whose velocity vector at p is v . For \mathbb{R}^n , under the Euclidean metric, since geodesics are given by straight lines, the exponential map is a simple addition: $\exp_p(v) = p + v$, for $p, v \in \mathbb{R}^n$. The exponential maps for more manifolds are listed in Section 4. The notions of tangent spaces, exponential maps, and their inverses are illustrated in Fig. 1.

3.1. Statistical analysis on manifolds

In this section, we briefly present the key ideas needed to perform statistical computations on manifolds. Take the case of the simplest statistic, the sample mean, for a sample set (x_1, x_2, \dots, x_n) on \mathbb{R}^n : $\bar{x}_k = \frac{1}{k} \sum_{i=1}^k x_i$. Now what if the underlying space is not \mathbb{R}^n but a non-Euclidean manifold instead? To answer this question we consider an n -dimensional Riemannian manifold M . Let $d(p, q)$ denote the length of the shortest geodesic between arbitrary points $p, q \in M$. We start the analysis by assuming that we are given a probability density function f on M . This function, by definition, satisfies the properties that $f : M \rightarrow \mathbb{R}_{\geq 0}$ and $\int_M f(p) dp = 1$, where dp denotes the reference measure on M with respect to which the density f is defined. There are two possibilities for computing statistics on M —intrinsic and extrinsic. The former uses the differential geometry of M to restrict the analysis on the manifold itself, while the latter embeds M inside a larger vector space, performs the analysis on the vector space, and projects the results back to M . The advantages and disadvantages of an extrinsic analysis are straightforward. The main advantage is its computational simplicity. Once an embedding is chosen, the rest of the analysis is quite standard and typically very fast. In contrast, the intrinsic analysis requires repeated computations of the exponential and inverse exponential maps. The disadvantage in extrinsic analysis is that the result depends on the choice of embedding which is often quite arbitrary. Also, in problems involving quotient

spaces, such as shape analysis, often there is no natural choice of the vector space for embedding the manifold.

A popular method for defining a mean on a manifold was proposed by Karcher (1977) who used the centroid of a density as its mean.

Definition 2. (Karcher mean) The Karcher mean μ_{int} of a probability density function f on M is defined as local minimizer of the cost function: $\rho : M \rightarrow \mathbb{R}_{\geq 0}$, where

$$\rho(p) = \int_M d(p, q)^2 f(q) dq, \quad (4)$$

dq denotes the reference measure used in defining the probability density f on M . The value of function ρ at the Karcher mean is called the *Karcher variance*. How does the definition of Karcher mean adapt to the sample set, i.e., a finite set of points drawn from an underlying probability distribution? Let q_1, q_2, \dots, q_k be independent random samples from the density f . Then, the sample Karcher mean of these points is defined to be the local minimizer of the function:

$$\rho_k(p) = \frac{1}{k} \sum_{i=1}^k d(p, q_i)^2. \quad (5)$$

An iterative algorithm for computing the sample Karcher mean is as follows. Let μ_0 be an initial estimate of the Karcher mean. Set $j = 0$:

- (1) For each $i = 1, \dots, k$, compute the tangent vector v_i such that the geodesic from μ_j , in the direction v_i , reaches q_i at time one, i.e., $\psi_1(\mu_j, v_i) = q_i$ or $v_i = \exp_{\mu_j}^{-1}(q_i)$.
- (2) Compute the average direction $\bar{v} = \frac{1}{k} \sum_{i=1}^k v_i$.
- (3) If $\|\bar{v}\|$ is small, then stop. Else, update μ_j in the update direction using $\mu_{j+1} = \psi_\epsilon(\mu_j, \bar{v})$, where $\epsilon > 0$ is small step size, typically 0.5. $\psi_t(p, v)$ denotes the geodesic path starting from p in the direction v parameterized by time t . In other words, $\mu_{j+1} = \exp_{\mu_j}(\epsilon \bar{v})$.
- (4) Set $j = j + 1$ and return to Step 1.

This algorithm is illustrated in Fig. 2. It can be shown that this algorithm converges to a local minimum of the cost function given in Eq. (5) which by definition is μ_{int} . Depending upon the initial value μ_0 and the step size ϵ , it converges to the nearest local minimum.

The tangent spaces of M provide a domain for defining covariances. For any point $p \in M$, let $p \rightarrow v \equiv \exp_{\mu}^{-1}(p)$ denote the inverse exponential map at μ from M to $T_{\mu}(M)$. The point μ maps to the origin $\mathbf{0} \in T_{\mu}(M)$ under this map. Now, we can define the Karcher covariance matrix as:

$$K_{int} = \int_{T_{\mu}(M)} vv^T f_v(v) dv, \quad v = \exp_{\mu}^{-1}(q),$$

where f_v is the induced probability density on the tangent space. For a finite sample set, the sample Karcher variance is given by $\tilde{K}_{int} = \frac{1}{k-1} \sum_{i=1}^k v_i v_i^T$, where $v_i = \exp_{\mu}^{-1}(q_i)$.

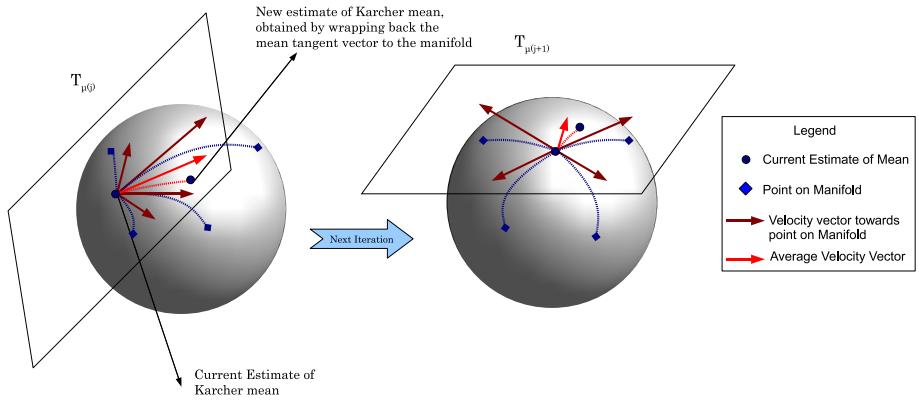


Fig. 2. Figure illustrating the Karcher mean computation algorithm. A set of points on the manifold is given. A random point from this set is chosen as the initial mean. Then, taking this mean as the pole, tangent vectors toward all other points are computed via the inverse exponential maps. A mean tangent vector is computed by straightforward averaging of tangent vectors. The mean tangent vector is wrapped onto the manifold by the exponential map. This new point forms the new estimate of the Karcher mean. The iterations continue in this manner.

4. Common manifolds arising in image analysis

In this section, we list a few commonly occurring manifolds in image and video understanding. We also list the required tools needed to perform statistical analysis such as tangent spaces, exponential maps, inverse exponential maps, etc. under some standard Riemannian metrics.

The hypersphere: The n -dimensional hypersphere, denoted by \mathbb{S}^n , can be shown to be a submanifold of \mathbb{R}^{n+1} . The tangent space at a point $p \in \mathbb{S}^n$, is just the orthogonal complement of $p \in \mathbb{R}^{n+1}$. Geodesics on a unit sphere \mathbb{S}^n are great circles (**Boothby, 1975**). Using the standard Riemannian metric, i.e., for any $v_1, v_2 \in T_p(\mathbb{S}^n)$, we use the Riemannian metric $\langle v_1, v_2 \rangle = v_1^T v_2$, the geodesics can be computed. The distance minimizing geodesic between two points p and q is the shorter of the two arcs of a great circle joining them between them. As a parameterized curve, this geodesic is given by

$$\alpha(t) = \frac{1}{\sin(\theta)} [\sin(\theta - t)p + \sin(t)q], \quad (6)$$

where $\theta = \cos^{-1}(\langle p, q \rangle)$.

The exponential map on a sphere, $\exp : T_p(\mathbb{S}^n) \mapsto \mathbb{S}^n$, is given by $\exp_p(v) = \cos(\|v\|)p + \sin(\|v\|)\frac{v}{\|v\|}$.

Special orthogonal group: The set of orthogonal matrices $O(n)$ is a subset of the manifold $GL(n)$ that satisfy the condition $OO^T = I$. Those orthogonal matrices with determinant +1 form the special orthogonal group, and denoted by $SO(n)$. One can show that the tangent space $T_O O(n) = \{OX | X \text{ is an } n \times n \text{ skew-symmetric matrix}\}$. Define the inner product for any $Y, Z \in T_O O(n)$ by $\langle Y, Z \rangle = \text{trace}(YZ^T)$, where trace denotes the sum of diagonal elements.

To define geodesics on $SO(n)$ with respect to the Riemannian metric defined above, we need the matrix exponential. For any $O \in SO(n)$ and any skew-symmetric matrix X , $\alpha(t) \equiv O\expm(tX)$ is the unique geodesic in $SO(n)$ passing through O with velocity OX at $t = 0$ (Boothby, 1975). The exponential maps for $SO(n)$ are given by $\exp_O(X) = O\expm(O^T X)$, and the inverse exponential maps are given by $\exp_{O_1}^{-1}(O_2) = O_1\logm(O_1^T O_2)$, where \expm and \logm refer to the matrix exponential and matrix logarithm, respectively.

Stiefel and Grassmann manifolds: The Stiefel and Grassmann manifolds are studied as quotient spaces of $SO(n)$. The Stiefel manifold $\mathcal{S}_{n,d}$ is the set of all d -dimensional orthogonal bases in \mathbb{R}^n , while the Grassmann manifold $\mathcal{G}_{n,d}$ is the space of d -dimensional subspaces of \mathbb{R}^n . Elements of $\mathcal{S}_{n,d}$ are denoted by $n \times d$ orthogonal matrix, i.e., $U \in \mathcal{S}_{n,d}$ implies $U \in \mathbb{R}^{n \times d}$ such that $U^T U = I_d$. The tangent space at any point U is

$$T_U(\mathcal{S}_{n,d}) = \left\{ O \begin{bmatrix} C \\ -B^T \end{bmatrix} \mid C = -C^T, C \in \mathbb{R}^{d \times d}, B \in \mathbb{R}^{d \times (n-d)} \right\}, \quad (7)$$

where $O = [UV]$ such that V is any arbitrary basis of the space perpendicular to U in \mathbb{R}^n . Similarly, elements of $\mathcal{G}_{n,d}$ are denoted by $[U] = \{UQ \mid Q \in SO(d)\}$ and the tangent space at any point $[U]$ is

$$T_{[U]}(\mathcal{G}_{n,d}) = \left\{ O \begin{bmatrix} 0 \\ -B^T \end{bmatrix} \mid B \in \mathbb{R}^{d \times (n-d)} \right\}, \quad (8)$$

and O is a completion of U as earlier. Geodesics in $\mathcal{S}_{n,d}$ and $\mathcal{G}_{n,d}$ can be realized as geodesics in the larger space $SO(n)$ as long as they are perpendicular to the corresponding orbits.

Symmetric positive definite matrices: The space of $d \times d$ symmetric positive definite (tensors/covariance matrices) is denoted as $Sym^+(d)$. The tangent space at any point X in $Sym^+(d)$ is given by the set of $d \times d$ symmetric matrices, i.e., $Sym(d)$. For a given point X , and any two tangent vectors $Y, Z \in T_X Sym^+(d)$, we use the inner product $\langle Y, Z \rangle_X = \text{trace}(X^{-1/2} Y X^{-1} Z X^{-1/2})$ (Pennec et al., 2006). Under this Riemannian metric, the geodesic passing through a point X in the direction specified by tangent vector W is given by $\gamma(t) = X^{1/2} \expm(tX^{-1/2} W X^{-1/2}) X^{1/2}$. The exponential map of a point $y \in T_X$ at X is given by

$$\exp_X(y) = X^{\frac{1}{2}} \expm\left(X^{-\frac{1}{2}} y X^{-\frac{1}{2}}\right) X^{\frac{1}{2}}, \quad (9)$$

and the inverse exponential map is given by

$$\exp_X^{-1}(Y) = X^{\frac{1}{2}} \logm\left(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}\right) X^{\frac{1}{2}}, \quad (10)$$

where the \expm and \logm refer to the matrix exponential and matrix logarithm, respectively.

5. Applications in image analysis

5.1. Manifolds in shape analysis

5.1.1. Landmark-based shape analysis

This approach for shape analysis was first proposed by Kendall (1984) and advanced by several others (Le and Kendall, 1993; Kendall et al., 1999; Small, 1996; Dryden and Mardia, 1998). The basic idea is to sample the object at a number of points, called *landmarks*, and form polygonal shapes by connecting those points with straight lines. Of course, the number and locations of these points on the objects can drastically change the resulting polygonal shapes but we will disregard that issue for the moment. One can organize these landmarks in the form of a vector of coordinates and perform standard vector calculus. Let $x \in \mathbb{R}^{n \times 2}$ represent n ordered points selected from the boundary of an object. It is often convenient to identify points in \mathbb{R}^2 with elements of \mathbb{C} , i.e., $x_i \equiv z_i = (x_{i,1} + jx_{i,2})$, where $j = \sqrt{-1}$. Thus, in this complex representation, a configuration of n points x is now $z \in \mathbb{C}^n$. As described earlier, one considers the joint action of the translation and the scaling group on the set of such configurations and forms an orthogonal section of that set. Each z is then represented by the corresponding element of the orthogonal section, often also called the *pre-shape space*: $\mathcal{D} = \{z \in \mathbb{C}^n | \frac{1}{n} \sum_{i=1}^n z_i = 0, \|z\| = 1\}$. \mathcal{D} is not a vector space because $a_1 z_1 + a_2 z_2$ for $a_1, a_2 \in \mathbb{R}$ and $z_1, z_2 \in \mathcal{D}$ is typically not in \mathcal{D} , due to the unit norm constraint. However, \mathcal{D} is a unit sphere and one can utilize the geometry of a sphere to analyze points on it. Under the Euclidean metric, the shortest path between any two elements $z_1, z_2 \in \mathcal{D}$, also called a *geodesic*, is given by the great circle: $\alpha_{ksa} : [0,1] \rightarrow \mathcal{D}$, where

$$\alpha_{ksa}(\tau) = \frac{1}{\sin(\theta)} [\sin(\theta(1-\tau))z_1 + \sin(\tau\theta)z_2] \quad \text{and} \quad \theta = \cos^{-1}(\Re(\langle z_1, z_2 \rangle)). \quad (11)$$

In order to compare the shapes represented by z_1 and z_2 , we need to align them rotationally. Let $[z]$ be the set of all rotations of a configuration z according to: $[z] = \{e^{j\phi} z | \phi \in \mathbb{S}^1\}$. One defines an equivalence relation on \mathcal{D} by setting all elements of this set as equivalent, i.e., $z_1 \sim z_2$ if there exists an angle ϕ such that $z_1 = e^{j\phi} z_2$. The set of all such equivalence classes is the quotient space $\mathcal{D}/U(1)$, where $U(1) = SO(2) = \mathbb{S}^1$ is the set of all rotations in \mathbb{R}^2 . This space is called the *complex projective space* and is denoted by \mathbb{CP}^{n-1} . A geodesic between two elements $z_1, z_2 \in \mathbb{CP}^{n-1}$ is given by computing α_{ksa} between z_1 and $e^{\phi^*} z_2$, where ϕ^* is the optimal rotational alignment of z_2 to z_1 . This rotational alignment is found using

$$\begin{aligned} \phi^* &= \underset{\phi \in \mathbb{S}^1}{\operatorname{argmin}} \|z_1 - e^{j\phi} z_2\|^2 \\ &= \underset{\phi \in \mathbb{S}^1}{\operatorname{argmax}} (\Re(e^{-j\phi} \langle z_1, z_2 \rangle)) = \theta, \quad \text{where } \langle z_1, z_2 \rangle = r e^{j\theta}. \end{aligned}$$

The length of the geodesic is given by θ and that quantifies the difference in shapes of the boundaries represented by z_1 and z_2 .

5.1.2. Planar curve-based shape analysis

In this section we consider methods that utilize the full curves, and not just a set of landmarks, for analyzing shapes of objects. These curves are typically boundaries or silhouettes of objects in images and it is natural to consider the shapes of full curves even if the data is sometimes noisy or corrupted. Let a parameterized planar curve be denoted as $\beta(t) \in \mathbb{R}^2$, where $t \in [0, 1]$ is a parameter. In case the curve is closed, it is more natural to parameterize using the domain \mathbb{S}^1 instead of $[0, 1]$ since $\beta(0) = \beta(1)$ and there is no natural point on the curve to call $\beta(0)$. Assuming that $\dot{\beta}(t) \neq 0$, we can decompose it in a polar form $\dot{\beta}(t) = s(t)\Theta(t)$, where $s(t) = \|\dot{\beta}(t)\| \in \mathbb{R}_+$ is the speed function and $\Theta(t) = \frac{\dot{\beta}(t)}{s(t)}$ denotes the unit vector in the direction of $\dot{\beta}(t)$. As in the previous section, one can use the complex notation to write $\Theta(t) = e^{j\theta(t)}$. Thus, $\theta(t)$ is the angle made by the velocity vector $\dot{\beta}(t)$ with respect to the x axis. If $s(t) = 1$ for all t , i.e., the curve is arc-length parameterized, then θ is a sufficient representation of the curve β for shape analysis. Another possibility is to use $\kappa(t) = \dot{\theta}(t)$, the curvature function along the curve, in shape analysis. Figure 3 shows these representations, coordinate functions, angle function, or curvature function, for a closed planar curve.

In case one re-parameterizes the given curves by arc-length, a comparison of their angle function leads to a so-called non-elastic shape analysis. This analysis is based on a linear registration of points across curves and simply comparing their angle functions, resulting in a metric that measures the bending energy from one curve to the other (Zahn and Roskies, 1972; Klassen et al., 2004). A more comprehensive solution is to allow arbitrary parameterizations of curves and solve for nonlinear registration of points across curves. This leads to an elastic analysis that was initiated by Younes (1998, 2008) and later developed for general Euclidean spaces by Mio et al. (2007) and Srivastava et al. (2010). The representation spaces of these methods are infinite-dimensional functional spaces since one needs full functions to represent continuous curves. In case of curves with unit lengths, the representations are restricted to functions with unit \mathbb{L}^2 norm, so the resulting space is a unit sphere in the Hilbert space \mathbb{L}^2 . In case we further restrict to only the closed curves, this additional constraint results in a submanifold of the Hilbert sphere whose geometry is more complicated and requires numerical techniques for computing geodesics and sample statistics.

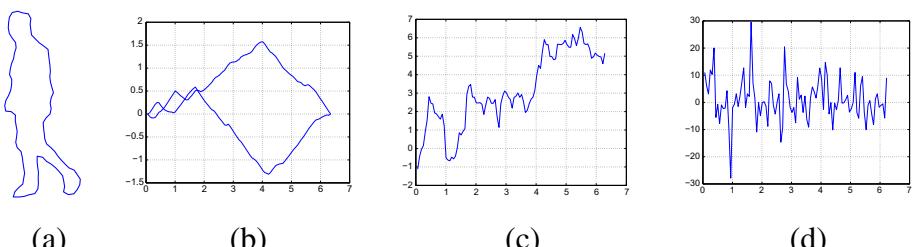


Fig. 3. (a) A simple closed curve β , (b) the two coordinate functions x and y plotted against the arc-length, (c) the angle function θ , and (d) the curvature function κ .

5.2. Manifolds in activity analysis

In vision-based human activity analysis, the goal is to identify what is happening in a scene given in a video of the event. In the context of single agent action analysis, e.g., walking, bending, running, throwing, etc. it has been found that mathematical representations of body shape and body movement are important cues toward this goal. It turns out that mathematical representations for these cues often lead to non-Euclidean spaces. As we have already seen, the notion of shape naturally leads to manifold interpretations. Further, as we will discuss in more detail in this section, mathematical models of motion also lead to analysis over non-Euclidean spaces. First, we shall start with reviewing some applications of manifold-based shape representations for human activity analysis. Next, we will discuss how a class of common parametric models to describe temporal variation of features leads to a Grassmann manifold interpretation. Then, we discuss how nonparametric representations of temporal variations in terms of warping functions again lead to manifold-valued constraints. Finally, we discuss a joint appearance and motion representation in terms of tensors and its associated manifold interpretation.

5.2.1. Shape features for activity analysis

As already discussed in earlier sections, there has been substantial work in shape representation for object recognition tasks. The shape of the human silhouette and its manner of variation has been shown to contain important information about the activity being performed. Thus, many of the shape representations discussed so far are naturally applicable to action recognition applications. However, they differ from object recognition applications due to the need to model time-series of shapes. One of the earliest applications of shape manifolds for action analysis is found in [Vaswani et al. \(2005\)](#) who applied Kendall's shape representation to model group activities in an airport surveillance setting. In their application, the shape formed by a group of humans embarking and disembarking from an aircraft was used to detect anomalous activities. This was achieved by modeling shape variation using an auto-regressive model on the tangent space at a mean shape. [Veeraraghavan \(2005\)](#) studied the utility of Kendall's shape representation for modeling single human actions. Given a binary image consisting of the silhouette of a person, the shape from this binary image is extracted. The procedure is graphically illustrated in Fig. 4a. To compare two shape-sequences, both nonparametric approaches (Dynamic-Time warping) and parametric approaches such as state-space modeling were considered. In the case of DTW, the Kendall's Procrustes distance is used during the DTW distance computations. For state-space modeling, the tangent space at a mean-shape is used to project data, then AR and ARMA model parameters are estimated on this tangent space. In cases, when one can extract more descriptive closed curve-based shape features, it can result in significantly better performance in applications like gesture recognition. In ([Abdelkader et al., 2011](#)), the closed curve representation of shapes is used to model gestures as a time-sequence on manifolds. A hidden Markov model is estimated using joint clustering and parametric density modeling for each cluster.

To illustrate the benefits obtained from manifold-based shape representations for activity recognition, we briefly present here some results from the USF database

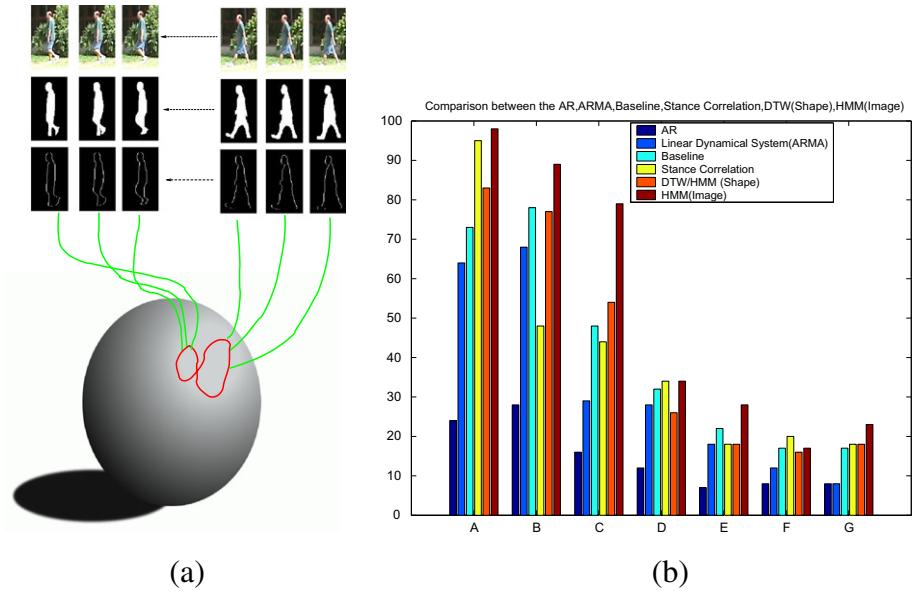


Fig. 4. (a) Graphical illustration of the sequence of shapes obtained during a walking cycle. (b) Bar diagram comparing the identification rate of various algorithms. Figure taken from [Veeraraghavan, \(2005\)](#).

taken from [Veeraraghavan, \(2005\)](#). Gait recognition experiments were designed for challenge experiments A–G. These experiments featured and tested the recognition performance against various covariates like the camera angle, shoe type, surface change, etc. ([Sarkar et al., 2005](#)). Figure 4b shows a comparison of the identification rate (rank 1) of the various shape and kinematics-based algorithms. It is observed that shape-based algorithms are comparable in performance to using general high-dimensional features such as entire background subtracted images themselves, and perform better than some common kinematics-based algorithms.

5.2.2. Grassmann manifold for human activity analysis

Once features are extracted from a video, a common approach to describe the temporal evolution/motion is by means of parametric state-space models. One such model is the linear dynamical system. A wide variety of high-dimensional time-series data such as dynamic textures, human joint angle trajectories, shape sequences, video-based face recognition, etc. have been modeled using such dynamical models ([Soatto et al., 2001; Bissacco et al., 2001; Veeraraghavan, 2005; Aggarwal, 2004](#)). Given training data the model parameters are estimated first and, during testing, one measures the similarity between the model parameters of the training and test instances. It was shown in [Turaga et al. \(2008, 2011\)](#) that the study of these models can be formulated as a study of the geometry of the Grassmann manifold. The linear dynamical model is given by

$$f(t) = Cz(t) + w(t) \quad w(t) \sim N(0, R), \quad (12)$$

$$z(t+1) = Az(t) + v(t) \quad v(t) \sim N(0, Q), \quad (13)$$

where z is the hidden state vector, A is the transition matrix, and C is the measurement matrix. f represents the observed features while w and v are noise components modeled as normal with 0 mean and covariance R and Q , respectively. Starting from an initial condition $z(0)$, it can be shown that the *expected* observation sequence is given by

$$E \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ \vdots \end{bmatrix} z(0) = O_\infty(M)z(0). \quad (14)$$

Thus, the expected observation sequence generated by a time-invariant model (A, C) lies in the column-space of the extended *observability* matrix given by $O_\infty = [C^T, (CA)^T, (CA^2)^T, \dots]^T$. However, motivated by the fact that human actions are of a finite-duration in time the study of the model can be simplified by considering only an n -length expected observation sequence instead of the infinite sequence as above. The n -length expected observation sequence generated by the model (A, C) lies in the column-space of the *finite* observability matrix given by

$$O_n^T = [C^T, (CA)^T, (CA^2)^T, \dots, (CA^{n-1})^T]. \quad (15)$$

Thus, a dynamical model can be identified by a point on the Grassmann manifold, corresponding to the subspace spanned by the columns of the finite observability matrix. Using the geometry of the manifold, one can devise nearest neighbor classifiers, parametric and nonparametric density models, etc. for classification (Turaga et al., 2011).

Here we discuss the application of this approach for summarizing videos containing human activities. Identifying visually salient aspects of long videos is an important application of human activity analysis. In activity-based summarization, the goal is to identify typical activities in a given domain of interest or in a given video. For this, one would first need to segment the video into coherent actions, and perform clustering on the obtained segments. However, since each segment can be of differing lengths, and possibly corrupted by noise, it is desirable to compactly parametrize each segment by a small set of parameters. For this parametrization, the linear dynamic model is proposed in Turaga et al. (2009). However, as discussed above, the space of these models is better understood as a Grassmann manifold. Thus, intrinsic clustering algorithms derived from the geometry of this space are needed. Here, we briefly present the approach reported in Turaga et al. (2009), in which color models of the foreground and background are used to segment the background and foreground pixels. After some post-processing, the final feature is a rescaled binary image of size 100×100 of the pixels inside the bounding box. Linear dynamic models of state-space dimension $d = 5$ are estimated for each segment. Then, a K-means clustering on the Grassmann manifold (Turaga et al., 2011) is used to obtain the clusters. Some sample sequences in the obtained clusters are shown in Fig. 5. The shown clusters correspond dominantly to “Sitting Spins” and “Standing Spins.”

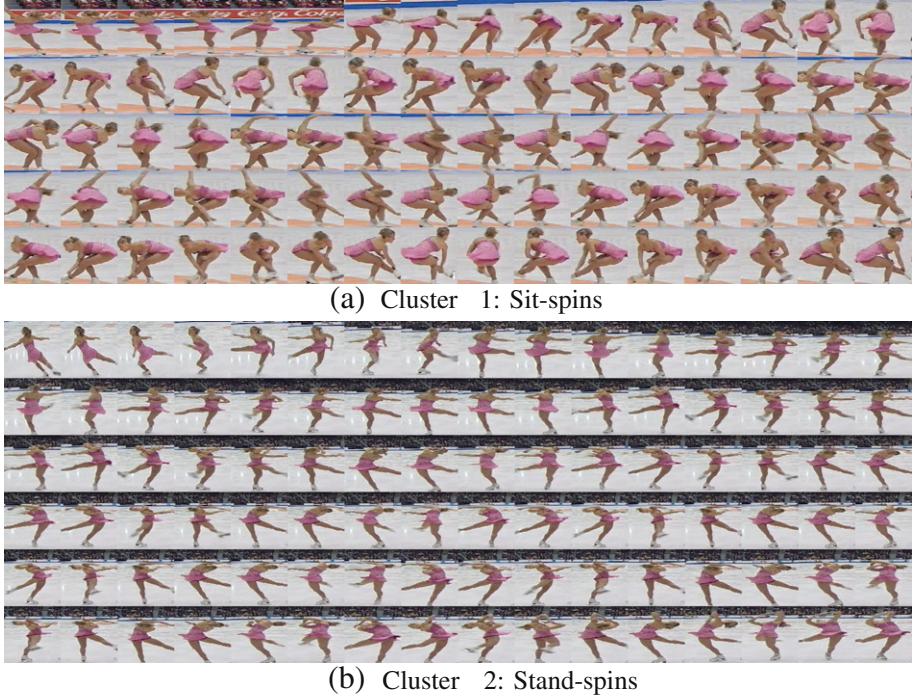


Fig. 5. Dynamical model-based clustering on the Grassmann manifold. Shown here are a few sequences from each obtained cluster from a long skating video. Each row in a cluster shows contiguous frames of a sequence. Figures taken from Turaga et al., (2009).

In Turaga and Chellappa (2009), a generalization of the dynamical model to a time-varying dynamical model is proposed, to model long videos which contain a long sequence of actions. In this approach, the model parameters (A, C, Q, R) are allowed to vary with time. Assuming that the model parameters change slowly with time, a local linear approximation to the time-varying process is constructed. This gives rise to a sequence of model parameters $M_t = (A_t, C_t)$. Each element in the sequence can be considered to be a point on the Grassmann manifold arising due to the time-varying observability matrix

$$O_n(M_t) = \left[\begin{array}{c} C_t; C_t A_t; \dots; C_t A_t^{n-1} \end{array} \right]. \quad (16)$$

To compactly represent the subspace variations, Turaga and Chellappa (2009) parametrize the trajectory using a switching model on the Grassmann manifold. This representation can be used to provide a visual summarization of long videos (Turaga and Chellappa, 2009). The clusters within each state represent the distinct actions and their variations in the video. The transition structure between the clusters represents how the overall activity in the video proceeds. Recent work has also focused on enabling fast search and retrieval for human actions based on dynamical models. By exploiting the Grassmann manifold formulation of dynamical models, Chaudhry and Ivanov (2010) and Turaga and Chellappa (2010)

designed approximate search methods using a hashing strategy. This was enabled by employing hashing techniques on the manifold's tangent spaces, by using the exponential and inverse exponential maps. This enables fast approximate search for actions in large databases in a computationally efficient manner. It was empirically shown that the tangent space approximation does not result in significant loss of accuracy in various databases.

5.2.3. Group activity modeling using trajectories

While video-based activity analysis and recognition has received much attention, a large body of existing work deals with single object/person case. Coordinated multi-object activities, or group activities, present in a variety of applications, such as surveillance, sports, and biological monitoring records, are important challenges. In far-field imaging conditions, where reasonable detection and tracking are possible, such as at an airport or a sports video, one can model group activities from the set of trajectories of individual objects.

However in co-ordinated actions, it is important to quantify interactions between objects. In [Li et al. \(2009\)](#), a view-stable and robust descriptor for group actions is proposed which is termed the Temporal Interaction Matrix. This matrix is constructed directly from tracking data, e.g., $Y(t_1, t_2, p_1, p_2)$ could refer to relative distances or relative speeds between person p_1 at time t_1 and person p_2 at time t_2 . The same framework is extended to cases where higher-resolution images are available for each person. In such a case, $Y(t_1, t_2, p_1, p_2)$ can be quantified in terms of the similarity between the features extracted from person at the corresponding time-instants. From the basic interaction matrix, a tensor-reduction operation is performed which results in a symmetric matrix with unit Frobenius norm. The set of such matrices is not a vector space, but a submanifold of the set of symmetric matrices. The geometric properties of this submanifold are used to design probabilistic classification algorithms. Multi-modal densities are estimated from training data and its utility is shown in recognizing various challenging group actions from American football videos.

5.2.4. Hilbert sphere for modeling execution-rate variations in activities

In addition to parametric time-series models, another approach has been to compare sequences of features by means of dynamic-time warping. In the context of activity recognition, different instances of the same activity may consist of varying relative speeds at which the actions are executed, in addition to other intra- and inter-person variabilities. Thus, it is important for algorithms for activity recognition to compensate for intra- and inter-personal variations of the same activity. Results on gait-based person identification shown in [Bobick and Tanawongsuwan \(2003\)](#) indicate that it is very important to take into account the temporal variations in the person's gait. In [Veeraraghavan \(2009\)](#), statistics of typical temporal warps for actions are used to further enhance activity templates. The basic idea is that the space of temporal warps is constrained for each action class, and a statistical characterization can be derived by formulating time warps as points on a Hilbert sphere.

Assume that for each frame of the video, an appropriate feature has been extracted and that the video data has now been converted into a feature sequence given by

f^1, f^2, \dots , for frames 1, 2, ..., respectively. In the following, \mathcal{F} is used to denote the feature space associated with the chosen feature. Let γ be a diffeomorphism (a diffeomorphism is a smooth, invertible function with a smooth inverse) from $[0, 1]$ to itself with $\gamma(0) = 0$ and $\gamma(1) = 1$. Also, let Γ be the set of all such functions. Elements of Γ are used to denote time-warping functions. The model for an activity now consists of an average activity sequence given by $a : [0, 1] \rightarrow \mathcal{F}$, a parameterized trajectory on the feature space. Any time-warped realization of this activity is then obtained using

$$r(t) = a(\gamma(t)), \quad \gamma \in \Gamma. \quad (17)$$

Equation (17) actually defines an action of Γ on $\mathcal{F}^{[0,1]}$, the space of all continuous activities. In this model, the variability associated with γ in each class is viewed as a distribution P_γ on Γ . By rewriting time-warping functions in square-root form, i.e., $\psi = +\sqrt{\gamma}$, a spherical manifold interpretation is provided.

Let the space of all square-root density forms be given by

$$\Psi = \left\{ \psi : [0, 1] \rightarrow \mathbb{R} | \psi \geq 0, \int_0^1 \psi^2(t) dt = 1 \right\}. \quad (18)$$

This is the positive orthant of a unit hypersphere in the Hilbert space of all square-integrable functions on $[0, 1]$. Since Ψ is a sphere, its geometry is well known. Using its geometry, algorithms for computing sample statistics, probability density functions, and generating inferences are developed in [Veeraraghavan \(2009\)](#). This framework provides methods to compare activities invariant to execution rates, and a generative model for actions by considering uncertainty in the execution rates.

5.2.5. Modeling activities as tensors: products of Grassmann manifolds

Tensors are generalizations of matrices to multiple dimensions. A 3D space–time volume can naturally be considered as a tensor with three independent dimensions. By decomposing a multi-dimensional data tensor into dominant modes (as a generalization of principal component analysis), one can extract signatures useful for action recognition. Using this approach, [Lui et al. \(2010\)](#) considered a tensor as a collection of subspaces. Each subspace is estimated by flattening the 3D tensor along one of the three directions, and then obtaining a low rank approximation to the flattened tensor via SVD. Thus, an (x, y, t) volume gives rise to three subspaces. Intuitively, these subspaces capture overall spatial appearance variations and motion variations along the horizontal and vertical directions. Each of the subspaces is identified as a point on the Grassmann manifold, and the tensor is viewed as a triplet of subspaces. This triplet is naturally considered as a point on a product of Grassmann manifolds. Geodesics on product manifolds can be computed simply as geodesics in individual component spaces. This approach is shown to outperform other tensor-based approaches such as canonical correlations and space–time interest point-based approaches on the challenging KTH dataset and YouTube action dataset.

5.3. Manifolds in face and gesture analysis

5.3.1. Face recognition using multiple images

In face recognition, recent methods have focused on utilizing multiple images of the same subject, taken under varying viewpoints or varying illumination conditions, for recognition (Kim et al., 2007; Hamm and Lee, 2008; Zhou and Chellappa, 2006). Since it was shown in Basri and Jacobs (2003) that the illumination cone of a convex Lambertian surface can be approximated by a nine-dimensional linear subspace, the set of face images of the same person under varying illumination conditions is frequently modeled as a linear subspace of nine-dimensions (Lee et al., 2005).

Given a large set of images indexed by, say, the pose or viewing angle of the camera, a view-dependent subspace is often used as the model of object appearance. The subspaces can be estimated by straightforward principal component analysis. Given another set of images during testing, the goal is to compute its most likely class. However, since subspaces are viewed as elements of a Grassmann manifold, the problem is reduced to learning a classifier over the Grassmann manifold. Toward this, Hamm and Lee (2008) performs discriminative classification over subspaces for object recognition tasks by using Mercer kernels on the Grassmann manifold. In Lui and Beveridge (2008), a face image and its perturbations due to registration errors are approximated as a linear subspace, hence are embedded as points on a Grassmann manifold. Turaga et al. (2011) uses parametric and nonparametric statistical models over the Grassmann manifold for face recognition from image sets. In all these formulations, the geometry of the Grassmann manifold was used to derive classification algorithms that outperform simpler classification methods.

5.3.2. Landmark-based facial aging and gesture analysis

The modeling of the appearance of human faces is an important component in several applications such as biometrics, animation, and picture annotation. Aging is a source of variation which has only recently been gaining attention. Understanding the appearance variations induced by aging is important for applications where the claimed identity and the enrolled face may show a large difference in apparent age. Studies in neuroscience have shown that facial geometry is a strong factor that influences age perception (O'Toole et al., 1999). Young faces exhibit distinct growth-related anthropometric trends. Anthropometric variations in adults are distinctive to a lesser degree than in children, but nevertheless they do exhibit drifts in facial features surrounding the mouth, eyebrows, etc. This is illustrated in Fig. 6 where distinct geometric changes can be observed as a person ages. By facial geometry we usually refer to the location of 2D facial landmarks on images.

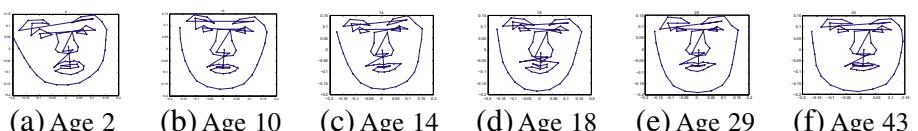


Fig. 6. Facial geometric variation across age. Samples shown correspond to individual 2 from the FG-net dataset. The geometry of the Grassmann manifold is used to estimate a regression function between shape of landmarks and age in Turaga et al., (2010).

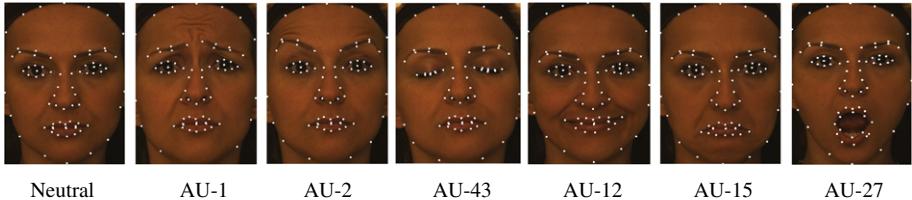


Fig. 7. A sequence from the Bosphorus database ([BIOID, 2008](#)) performing various facial action units. The landmark locations are shown on the faces. The face shape is viewed as a point on the Grassmann manifold, and the expression is seen as a curve on the Grassmann manifold. Figure taken from [Taheri et al., \(2011\)](#).

The *affine shape space* ([Goodall and Mardia, 1999](#)) is useful to account for small changes in camera location or change in the pose of the subject. The affine transforms of the shape can be derived from the base shape simply by multiplying the shape matrix L by a 2×2 full rank matrix on the right. For example, let A be a 2×2 affine transformation matrix i.e., $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$. Then, all affine transforms of the

base shape L_{base} can be expressed as $L_{\text{affine}}(A) = L_{\text{base}}A^T$. Note that, multiplication by a full-rank matrix on the right preserves the column-space of the matrix L_{base} . Thus, the 2D subspace of \mathbb{R}^m spanned by the columns of the matrix L_{base} is an *affine-invariant* representation of the shape. i.e., $\text{span}(L_{\text{base}})$ is invariant to affine transforms of the shape. Subspaces such as these can be identified as points on a Grassmann manifold.

In [Turaga et al. \(2010\)](#), the geometry of the Grassmann manifold is used to estimate a regression function that relates facial geometry to the age of the person. The regression function was estimated on the tangent space at the dataset mean. In [Taheri et al. \(2011\)](#), facial expressions were analyzed using landmarks on faces (Fig. 7). Each facial landmark configuration was viewed as a point on the Grassmann manifold, and the expression was seen as a sequence on the Grassmann manifold. Sequences were parametrized by a sequence of tangent vectors, which were parallel transported to a common tangent space. These parallel-transported tangent vectors were used for expression recognition.

5.4. Manifolds in object detection, recognition, and tracking

5.4.1. Covariance-based detection and tracking

In recent years, region covariance has proved to be a popular feature to encode local shape and texture from images. Covariance features as region descriptors were introduced in [Tuzel et al. \(2006\)](#) and have been successfully applied to human detection ([Tuzel et al., 2008](#)), object tracking ([Porikli et al., 2006](#)), and texture classification ([Tuzel et al., 2006](#)). In their approach, given a rectangular image region, several simple gradient-based features are computed at each pixel location e.g., $F(x,y) = \left[I(x,y), \left| \frac{\partial I}{\partial x} \right|, \left| \frac{\partial I}{\partial y} \right|, \left| \frac{\partial^2 I}{\partial x^2} \right|, \left| \frac{\partial^2 I}{\partial y^2} \right| \right]^T$, where (x,y) denotes the pixel location in image I . Then, from the set of such d -dimensional vectors (in the above example

$d = 5$) the $d \times d$ covariance matrix of these features is computed as the overall descriptor for the image region.

This approach was used for the texture classification problem in Tuzel et al. (2006). For each texture image, several rectangular regions are randomly sampled, and each region represented by the covariance of features in that region. The entire image is thus represented by the set of all such covariance features. Recognition is performed using a majority voting rule. A similar approach is used for the problem of pedestrian detection from images in Tuzel et al. (2008). In this approach, the covariance matrix representation is used to train a weak-classifier for pedestrian detection. Each weak classifier corresponds to a sub-region of the original image, and each weak classifier is trained in the tangent space of the mean covariance descriptor for that region. Several such weak classifiers are combined in the boosting framework to obtain a high accuracy classifier. The covariance representation of image regions is used in a tracking framework in Porikli et al. (2006), where object templates are represented as covariance matrices, and the geodesic distance is used to identify candidate regions in a new image. Template models are updated over time using a simple intrinsic mean computation.

5.4.2. 3D-Model based-detection and spatio-temporal alignment

One of the earlier applications of manifold-based statistical analysis in computer vision was in detecting, tracking, and recognizing 3D objects of interest in 2D images. The basic approach here was to model the image data as (Srivastava et al., 2005):

$$I = \mathcal{P}(g \cdot I^\alpha) + \text{noise},$$

where

- (1) I^α is a 3D (e.g., CAD) model of an object labeled by $\alpha \in \mathcal{A}$.
- (2) $g \in G$ is a transformation generally consisting of rotation, translation, and scale.

In the problems relating to pose estimation $G = SO(3)$, tracking $G = \mathbb{R}^3$, and pose and location tracking $G = SE(3)$.

- (3) \mathcal{P} is a projection of the 3D scene containing the target into the 2D image space. Depending upon applications, one can use either the orthographic or perspective projection here.

Using specific choices of models for noise and g , one can define posterior probability on either the transformation g , given by $P(g|I, \alpha)$, or the object label α , given by $P(\alpha|I) = P(\alpha) \int_G P(g|I, \alpha) dg$. Then, one can use Monte Carlo methods (Miller et al., 1995; Srivastava et al., 2002, 1997) or asymptotic ideas (Grenander et al., 2000) for generating Bayesian inferences for the detection, tracking, and recognition problems. These formulations require setting up manifold-valued estimators and analysis of their properties, such as consistency and efficiency (Grenander et al., 1998). In fact, Miller et al. (1995) described a Jump-Diffusion algorithm for filtering on nonlinear manifolds that was a precursor for the celebrated particle filtering method.

A similar problem is encountered in spatio-temporal alignment, where one tries to find an optimal transformation so that two given images or videos, or features

computed from them, are brought into correspondence. The likelihood function in these cases is usually in the form of a distortion measure on image and video features or pixels themselves. The spatial aspect of the misalignment is usually modeled by parametric transforms, including affine, homography, and projective, between the image plane coordinates of the two signals, based on different assumptions made regarding imaging conditions. The temporal misalignment, on the other hand, has traditionally been viewed as differences in frame rate or shift synchronization, which are modeled as a 1 D affine transform along the time axis. In [Li and Chellappa \(2010\)](#), temporal misalignment is generalized beyond just the camera-related factors (frame rate and temporal shift), to include more meaningful factors such as the observed dynamics. Temporal transformations of human activities as already discussed earlier can be highly non-rigid, and are better seen as elements of a diffeomorphism group. This was used in [\(Li and Chellappa, 2010\)](#) to derive a framework to perform video-to-video alignment using sequential importance sampling on manifolds. In this framework, registration is essentially an optimization problem over certain groups e.g., Euclidean, affine, etc. Optimization algorithms can then be extended to manifolds using many of the same tools discussed earlier. A good compilation of tools for performing gradient based optimization can be found in [Absil et al. \(2008\)](#). In addition to gradient-based techniques, one can easily employ Monte Carlo techniques on manifolds for these tasks.

6. Summary and discussion

In this chapter, we have presented an overview of recent advances in image and video analysis techniques that involve non-Euclidean geometry. In all these applications, the manifold structure arises due to invariance requirements or modeling assumptions. Once the geometry of these spaces is well understood mathematically, it paves the way toward employing intrinsic geometric statistics, and optimization tools. The results of such an analysis are often more meaningful than a corresponding Euclidean analysis, and result in significant improvements in performance.

Acknowledgments

This work was partially supported by the ONR MURI Grant and the ONR Grant N000140910664.

References

- Abdelkader, M.F., Abd-Almageed, W., Srivastava, A., Chellappa, R., 2011. Silhouette-based gesture and action recognition via modeling trajectories on Riemannian shape manifolds. *Comput. Vis. Image Understand.* 115, 439–455.
- Bosphorus database for 3D face analysis, 2008. In: BIOID.
- Absil, P.-A., Mahony, R., Sepulchre, R., 2008. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ.
- Aggarwal, G., Roy-Chowdhury, A., Chellappa, R., 2004. A system identification approach for video-based face recognition. In: *International Conference on Pattern Recognition*.

- Amit, Y., Grenander, U., Piccioni, M., 1991. Structural image restoration through deformable templates. *J. Am. Stat. Assoc.*
- Basri, R., Jacobs, D.W., 2003. Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.* 25 (2), 218–233.
- Begelfor, E., Werman, M., 2006. Affine invariance revisited. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 2087–2094.
- Bissacco, A., Chiuso, A., Ma, Y., Soatto, S., 2001. Recognition of human gaits. In: IEEE International Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 52–57.
- Bobick, A., Tanawongsuwan, R., 2003. Performance analysis of time-distance gait parameters under different speeds. In: Audio- and Video-Based Biometric Person Authentication (AVBPA), June.
- Boothby, W.M., 1975. An Introduction to Differentiable Manifolds and Riemannian Geometry. Academic Press Inc.
- Chaudhry, R., Ivanov, Y., 2010. Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos. In: European Conference on Computer Vision, pp. 735–748.
- Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 886–893.
- Dryden, I.L., Mardia, K.V., 1998. Statistical Shape Analysis. John Wiley & Son.
- Georghiades, A. S., Belhumeur, P. N., Kriegman, D. J., 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (6), 643–660.
- Goodall, C.R., Mardia, K.V., 1999. Projective shape analysis. *J. Comput. Graph. Stat.* 8 (2).
- Grenander, U., 1976. Pattern Synthesis: Lectures in Pattern Theory, vol. I, II. Springer-Verlag, New York, p. 1978.
- Grenander, U., 1981. Regular Structures: Lectures in Pattern Theory, vol. III. Springer-Verlag, New York.
- Grenander, U., 1993. General Pattern Theory. Oxford University Press.
- Grenander, U., Chow, Y., Keenan, D., 1990. HANDS: A Pattern Theoretic Study of Biological Shapes. Springer-Verlag.
- Grenander, U., Miller, M.I., 1994. Representations of knowledge in complex systems. *J. Roy. Stat. Soc. Ser. B* 56 (3).
- Grenander, U., Miller, M.I., 1998. Computational anatomy: an emerging discipline. *Quart. Appl. Math. L VI* (4), 617–694.
- Grenander, U., Miller, M.I., Srivastava, A., 1998. Hilbert–Schmidt lower bounds for estimators on matrix Lie groups for ATR. *IEEE Trans. PAMI* 20 (8), 790–802.
- Grenander, U., Srivastava, A., Miller, M.I., 2000. Asymptotic performance analysis of Bayesian object recognition. *IEEE Trans. Inform. Theory* 46 (4), 1658–1666.
- Hafner, J., Sawhney, H.S., Equitz, W., Flickner, M., Niblack, W., 1995. Efficient color histogram indexing for quadratic form distance functions. *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (7), 729–736.
- Hamm, J., Lee, D.D., 2008. Grassmann discriminant analysis: a unifying view on subspace-based learning. In: International Conference on Machine Learning, Helsinki, Finland, June, pp. 376–383.
- Kanatani, K., 1990. Group-Theoretical Methods in Image Understanding. Springer-Verlag.
- Karcher, H., 1977. Riemannian center of mass and mollifier smoothing. *Comm. Pure Appl. Math.* 30 (5), 509–541.
- Kendall, D.G., Barden, D., Carne, T.K., Le, H., 1999. Shape and Shape Theory. Wiley.
- Kendall, D.G., 1984. Shape manifolds, procrustean metrics and complex projective spaces. *Bull. Lond. Math. Soc.* 16, 81–121.
- Kim, T.K., Kittler, J., Cipolla, R., 2007. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (6), 1005–1018.
- Klassen, E., Srivastava, A., Mio, W., Joshi, S.H., 2004. Analysis of planar shapes using geodesic paths on shape spaces. *IEEE Trans. Pattern Anal. Mach. Intell.* 26 (3), 372–383.
- Le, H.L., Kendall, D.G., 1993. The Riemannian structure of Euclidean shape spaces: a novel environment for statistics. *Ann. Stat.* 21 (3), 1225–1271.

- Lee, K.-C., Ho, J., Kriegman, D.J., 2005. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (5), 684–698.
- Li, R., Chellappa, R., 2010. Aligning spatio-temporal signals on a special manifold. In: *ECCV*, vol. 5, pp. 547–560.
- Li, R., Chellappa, R., Zhou, S., 2009. Learning multi-modal densities on discriminative temporal interaction manifold for group activity recognition. In: *IEEE International Conference on Computer Vision and Pattern Recognition*.
- Lowe, D., 2003. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* 20.
- Lui, Y.M., Beveridge, J.R., 2008. Grassmann registration manifolds for face recognition. In: *European Conference on Computer Vision*, October, Marseille, France, pp. 44–57.
- Lui, Y.M., Beveridge, J.R., Kirby, M., 2010. Action classification on product manifolds. In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 833–839.
- Michor, P.W., Mumford, D., 2006. Riemannian geometries on spaces of plane curves. *J. Eur. Math. Soc.* 8, 1–48.
- Miller, M.I., Srivastava, A., Grenander, U., 1995. Conditional-expectation estimation via jump-diffusion processes in multiple target tracking/recognition. *IEEE Trans. Signal Process.* 43 (11) 2678–2690.
- Miller, M.I., Christensen, G.E., Amit, Y., Grenander, U., 1993. Mathematical textbook of deformable neuroanatomies. *Proc. Natl. Acad. Sci.* 90 (24).
- Mio, W., Srivastava, A., Joshi, S.H., 2007. On shape of plane elastic curves. *Int. J. Comput. Vis.* 73 (3), 307–324.
- Moakher, M., 2002. Means and averaging in the group of rotations. *SIAM J. Matrix Anal. Appl.* 24.
- O'Toole, A.J., Price, T., Vetter, T., Bartlett, J.C., Blanz, V., 1999. Three-dimensional shape and two-dimensional surface textures of human faces: the role of averages in attractiveness and age. *Image Vis. Comput.* 18 (1), 9–19.
- Pennec, X., Fillard, P., Ayache, N., 2006. A Riemannian framework for tensor computing. *Int. J. Comput. Vis.* 66 (1), 41–66.
- Porikli, F., Tuzel, O., Meer, P., 2006. Covariance tracking using model update based on Lie algebra. In: *IEEE International Conference on Computer Vision and Pattern Recognition*, June, New York, USA, pp. 728–735.
- Sarkar, S., Phillips, P.J., Liu, Z., Vega, I.R., Grother, P., Bowyer, K.W., 2005. The humanID gait challenge problem: data sets, performance, and analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 162–177.
- Small, C.G., 1996. *The Statistical Theory of Shape*. Springer.
- Soatto, S., Doretto, G., Wu, Y.N., 2001. Dynamic textures. In: *IEEE International Conference on Computer Vision*, vol. 2, pp. 439–446.
- Srivastava, A., Grenander, U., Jensen, G.R., Miller, M.I., 2002. Jump-diffusion Markov processes on orthogonal groups for object recognition. *J. Stat. Plan. Infer.* 103 (1–2), 15–37.
- Srivastava, A., Klassen, E., Joshi, S.H., Jermyn, I.H., in press. Shape analysis of elastic curves in euclidean spaces. *IEEE Trans. Pattern Anal. Mach. Intell.* <http://dx.doi.org/10.1109/TPAMI.2010.184>.
- Srivastava, A., Miller, M.I., Grenander, U., 1997. Ergodic algorithms on special euclidean groups for ATR. In: *Systems and Control in the Twenty-First Century: Progress in Systems and Control*, vol. 22, Birkhauser.
- Srivastava, A., Miller, M.I., Grenander, U., 2005. Statistical models for Bayesian object recognition. In: *Handbook of Image and Video Processing (Communications, Networking and Multimedia)*. Academic Press, pp. 1341–1354.
- Taheri, S., Turaga, P.K., Chellappa, R., 2011. Towards view-invariant expression analysis using analytic shape manifolds. In: *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*.
- Turaga, P.K., Chellappa, R., 2009. Locally time-invariant models of human activities using trajectories on the Grassmannian. In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 2435–2441.
- Turaga, P.K., Chellappa, R., 2010. Nearest-neighbor algorithms on non-Euclidean manifolds for computer vision applications. In: *Indian Conference on Computer Vision, Graphics, and Image Processing (ICVGIP)*.
- Turaga, P.K., Veeraghavan, A., Chellappa, R., 2009. Unsupervised view and rate invariant clustering of video sequences. *Comput. Vis. Image Understand.* 113 (3), 353–371.

- Turaga, P.K., Veeraraghavan, A., Srivastava, A., Chellappa, R., 2011. Statistical computations on Grassmann and Stiefel manifolds for image and video based recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (11), 2273–2286.
- Turaga, P.K., Veeraraghavan, A., Chellappa, R., 2008. Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision. In: *IEEE International Conference on Computer Vision and Pattern Recognition*.
- Turaga, P.K., Soma, B., Rama, C., 2010. The role of geometry in age estimation. In: *ICASSP*, pp. 946–949.
- Tuzel, O., Porikli, F., Meer, P., 2006. Region covariance: a fast descriptor for detection and classification. In: *European Conference on Computer Vision*, Graz, Austria, May, pp. 589–600.
- Tuzel, O., Porikli, F., Meer, P., 2008. Pedestrian detection via classification on Riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (10), 1713–1727.
- Vaswani, N., Roy Chowdhury, A.K., Chellappa, R., 2005. Shape activity: a continuous-state HMM for moving/deforming shapes with application to abnormal activity detection. *IEEE Trans. Image Process.* 14 (10), 1603–1616.
- Veeraraghavan, A., Roy-Chowdhury, A., Chellappa, R., 2005. Matching shape sequences in video with an application to human movement analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (12), 1896–1909.
- Veeraraghavan, A., Srivastava, A., Roy Chowdhury, A.K., Chellappa, R., 2009. Rate-invariant recognition of humans and their activities. *IEEE Trans. Image Process.* 18 (6), 1326–1339.
- Younes, L., 1998. Computable elastic distance between shapes. *SIAM J. Appl. Math.* 58 (2), 565–586.
- Younes, L., Michor, P.W., Shah, J., Mumford, D., Lincei, R., 2008. A metric on shape space with explicit geodesics, *Matematica E Applicazioni* 19 (1), 25–57.
- Zahn, C.T., Roskies, R.Z., 1972. Fourier descriptors for plane closed curves. *IEEE Trans. Comput.* 21 (3).
- Zhou, S.K., Chellappa, R., 2006. From sample similarity to ensemble similarity: probabilistic distance measures in reproducing kernel Hilbert space. *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (6), 917–929.

This page is intentionally left blank

Dictionary-Based Methods for Object Recognition*

Vishal M. Patel¹ and Rama Chellappa²

¹*Center for Automation Research, UMIACS, University of Maryland,
College Park, MD, USA*

²*Department of Electrical and Computer Engineering and the Center for
Automation Research, UMIACS, University of Maryland, College Park, MD,
USA*

Abstract

In recent years, Sparse Representation (SR) and Dictionary Learning (DL) have emerged as powerful tools for efficient processing image and video data in non-traditional ways. An area of promise for these theories is object recognition. In this chapter, we review the role of algorithms based on SR and DL for object recognition.

Keywords: sparse representation, dictionary learning, object recognition, biometrics recognition, discriminative dictionary learning

1. Introduction

Sparse and redundant signal representations have recently drawn much interest in vision, signal, and image processing (Elad et al., 2010; Wright et al., 2010; Rubinstein et al., 2010; Patel and Chellappa, 2010). This is due in part to the fact that signals and images of interest can be sparse or compressible in some dictionary. The dictionary can be either based on a mathematical model of the data or it can be learned directly from the data. It has been observed that learning a dictionary directly from training data rather than using a predetermined dictionary such as wavelet or Fourier usually leads to better representation and hence can provide improved results in many practical applications such as image restoration and classification (Rubinstein et al., 2010; Wright et al., 2010; Patel and Chellappa, 2010).

In this chapter, we summarize approaches to object recognition based on sparse representation (SR) and dictionary learning (DL). We first highlight the idea

*This work was partially supported by an ONR grant N00014–12-1-0124.

behind sparse representation. Then, we will outline the Sparse Representation-based Classification (SRC) algorithm (Wright et al., 2009) and present its applications in robust biometrics recognition (Wright et al., 2009; Pillai et al., 2009, 2011). Finally, we present several supervised, unsupervised, semi-supervised, and weakly supervised dictionary learning algorithms for object representation and recognition.

2. Sparse representation

Representing a signal involves the choice of a basis, where the signal is uniquely represented as a linear combination of the basis elements. In the case when we use an orthogonal basis, the representation coefficients are simply found by computing inner products of the signal with the basis elements. In the non-orthogonal basis case, the coefficients are found by taking the inner products of the signal with the bi-orthogonal basis. Due to the limitations of orthogonal and bi-orthogonal basis in representing complex signals, overcomplete dictionaries have been used. An overcomplete dictionary has more elements, also known as atoms, than the dimension of the signal.

Consider the dictionary $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_L] \in \mathbb{R}^{N \times L}$, where $L \geq N$ and the columns of \mathbf{B} are the dictionary atoms. Representing $\mathbf{x} \in \mathbb{R}^N$ using \mathbf{B} entails solving the following optimization problem

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}'}{\operatorname{argmin}} C(\boldsymbol{\alpha}') \quad \text{subject to } \mathbf{x} = \mathbf{B}\boldsymbol{\alpha}' \quad (1)$$

for some cost function $C(\boldsymbol{\alpha})$. In practice, since we want the sorted coefficients to decay quickly, sparsity of the representation is usually enforced. This can be done by choosing, $C(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_p, 0 \leq p \leq 1$, where $\|\cdot\|_p$ denotes the ℓ_p -norm defined as

$$\|\mathbf{x}\|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}}$$

and the ℓ_0 -norm is defined as the limit as $p \rightarrow 0$ of the ℓ_p -norm

$$\|\mathbf{x}\|_0 = \lim_{p \rightarrow 0} \|\mathbf{x}\|_p^p = \lim_{p \rightarrow 0} \sum_i |x_i|^p.$$

In general, the ℓ_0 -norm counts the number of non-zero elements in a vector

$$\|\mathbf{x}\|_0 = \#\{i : x_i \neq 0\}. \quad (2)$$

Figure 1 shows the behavior of the scalar weight functions $|\alpha|^p$ for various values of p . Note that as p goes to zero, $|\alpha|^p$ becomes a count of the non-zeros in $\boldsymbol{\alpha}$. Hence, by setting $C(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_0$, one can look for the sparsest solution to the underdetermined linear system of equations $\mathbf{x} = \mathbf{B}\boldsymbol{\alpha}$. The optimization problem in this case becomes the following:

$$(P_0) \quad \hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}'}{\operatorname{argmin}} \|\boldsymbol{\alpha}'\|_0 \quad \text{subject to } \mathbf{x} = \mathbf{B}\boldsymbol{\alpha}'. \quad (3)$$

As it turns out, this problem is NP-hard and cannot be solved in polynomial time. As a result, other alternatives are usually sought. The approach often taken in practice

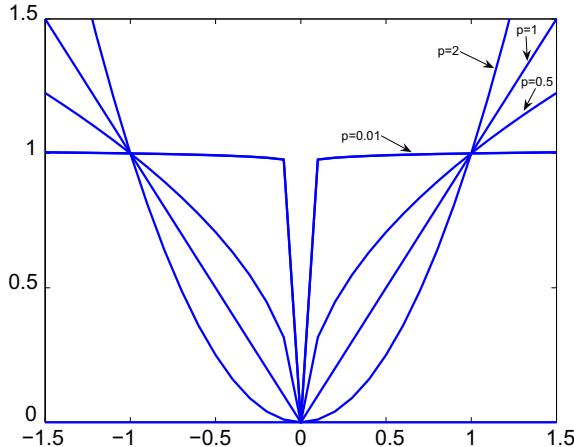


Fig. 1. The behavior of the scalar function $|x|^p$ for various values of p . As p goes to zeros, $|x|^p$ becomes the delta function, which is zeros for $x = 0$ and 1 elsewhere.

is to instead solve the following ℓ_1 -minimization problem

$$(P_1) \hat{\boldsymbol{\alpha}} = \operatorname{argmin}_{\boldsymbol{\alpha}'} \|\boldsymbol{\alpha}'\|_1 \quad \text{subject to } \mathbf{x} = \mathbf{B}\boldsymbol{\alpha}'. \quad (4)$$

(P_1) is a convex optimization problem and it is the one closest to (P_0) which can be solved by the standard optimization tools. Problem (4) is often referred to as Basis Pursuit (Chen et al., 1998). It has been shown that for \mathbf{B} with incoherent columns, whenever, (P_0) has a sufficiently sparse solution that solution is unique and is equal to the solution of (P_1) .

Define the mutual coherence of the matrix \mathbf{B} as follows:

Definition 2.1 (Bruckstein et al., 2009) The mutual coherence of a given matrix \mathbf{B} is the largest absolute normalized inner product between different columns from \mathbf{B} . Denoting the k th column in \mathbf{B} by \mathbf{b}_k , the mutual coherence is given by

$$\tilde{\mu}(\mathbf{B}) = \max_{1 \leq k, j \leq L, k \neq j} \frac{|\mathbf{b}_k^T \mathbf{b}_j|}{\|\mathbf{b}_k\|_2 \cdot \|\mathbf{b}_j\|_2}.$$

With this definition, one can state the following theorem:

Theorem 2.1 (Donoho and Elad, 2003; Gribonval and Nielsen, 2003) For the system of linear equations $\mathbf{x} = \mathbf{B}\boldsymbol{\alpha}$ ($\mathbf{B} \in \mathbb{R}^{N \times L}$ full rank with $L \geq N$), if a solution $\boldsymbol{\alpha}$ exists obeying

$$\|\boldsymbol{\alpha}\|_0 < \frac{1}{2} \left(1 + \frac{1}{\tilde{\mu}(\mathbf{B})} \right)$$

that solution is both unique solution of (P_1) and the unique solution of (P_0) .

In the rest of the section, we will show how variants of (1) can be used to develop robust algorithms for object classification.

2.1. Sparse representation-based classification

In object recognition, given a set of labeled training samples, the task is to identify the class to which a test sample belongs to. Following Wright et al., (2009), we briefly describe the use of sparse representations for biometric recognition, however, this framework can be applied to a general object recognition problem.

Suppose that we are given L distinct classes and a set of n training images per class. One can extract an N -dimensional vector of features from each of these images. Let $\mathbf{B}_k = [\mathbf{x}_{k1}, \dots, \mathbf{x}_{kj}, \dots, \mathbf{x}_{kn}]$ be an $N \times n$ matrix of features from the k th class, where \mathbf{x}_{kj} denote the feature from the j th training image of the k th class. Define a new matrix or dictionary \mathbf{B} , as the concatenation of training samples from all the classes as

$$\begin{aligned}\mathbf{B} &= [\mathbf{B}_1, \dots, \mathbf{B}_L] \in \mathbb{R}^{N \times (nL)} \\ &= [\mathbf{x}_{11}, \dots, \mathbf{x}_{1n} | \mathbf{x}_{21}, \dots, \mathbf{x}_{2n} | \dots \dots | \mathbf{x}_{L1}, \dots, \mathbf{x}_{Ln}].\end{aligned}$$

We consider an observation vector $\mathbf{y} \in \mathbb{R}^N$ of unknown class as a linear combination of the training vectors as

$$\mathbf{y} = \sum_{i=1}^L \sum_{j=1}^n \alpha_{ij} \mathbf{x}_{ij} \quad (5)$$

with coefficients $\alpha_{ij} \in \mathbb{R}$. The above equation can be written more compactly as

$$\mathbf{y} = \mathbf{B}\boldsymbol{\alpha}, \quad (6)$$

where

$$\boldsymbol{\alpha} = [\alpha_{11}, \dots, \alpha_{1n} | \alpha_{21}, \dots, \alpha_{2n} | \dots \dots | \alpha_{L1}, \dots, \alpha_{Ln}]^T \quad (7)$$

and \cdot^T denotes the transposition operation. We assume that given sufficient training samples of the k th class, \mathbf{B}_k , any new test image $\mathbf{y} \in \mathbb{R}^N$ that belongs to the same class will lie approximately in the linear span of the training samples from the class k . This implies that most of the coefficients not associated with class k in (7) will be close to zero. Hence, $\boldsymbol{\alpha}$ is a sparse vector.

In order to represent an observed vector $\mathbf{y} \in \mathbb{R}^N$ as a sparse vector $\boldsymbol{\alpha}$, one needs to solve the system of linear Eq. (6). Typically $L \cdot n \gg N$ and hence the system of linear Eq. (6) is underdetermined and has no unique solution. As mentioned earlier, if $\boldsymbol{\alpha}$ is sparse enough and \mathbf{B} satisfies certain properties, then the sparsest $\boldsymbol{\alpha}$ can be recovered by solving the following optimization problem:

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}'}{\operatorname{argmin}} \|\boldsymbol{\alpha}'\|_1 \text{ subject to } \mathbf{y} = \mathbf{B}\boldsymbol{\alpha}'. \quad (8)$$

When noisy observations are given, Basis Pursuit DeNoising (BP DN) can be used to approximate $\boldsymbol{\alpha}$

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}'}{\operatorname{argmin}} \|\boldsymbol{\alpha}'\|_1 \text{ subject to } \|\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}'\|_2 \leq \varepsilon, \quad (9)$$

where we have assumed that the observations are of the following form:

$$\mathbf{y} = \mathbf{B}\boldsymbol{\alpha} + \boldsymbol{\eta} \quad (10)$$

with $\|\boldsymbol{\eta}\|_2 \leq \varepsilon$.

Given an observation vector \mathbf{y} from one of the L classes in the training set, one can compute its coefficients $\hat{\boldsymbol{\alpha}}$ by solving either (8) or (9). One can perform classification based on the fact that high values of the coefficients $\hat{\boldsymbol{\alpha}}$ will be associated with the columns of \mathbf{B} from a single class. This can be done by comparing how well the different parts of the estimated coefficients, $\hat{\boldsymbol{\alpha}}$, represent \mathbf{y} . The minimum of the representation error or the residual error can then be used to identify the correct class. The residual error of class k is calculated by keeping the coefficients associated with that class and setting the coefficients not associated with class k to zero. This can be done by introducing a characteristic function, $\Pi_k : \mathbb{R}^n \rightarrow \mathbb{R}^n$, that selects the coefficients associated with the k th class as follows:

$$r_k(\mathbf{y}) = \|\mathbf{y} - \mathbf{B}\Pi_k(\hat{\boldsymbol{\alpha}})\|_2. \quad (11)$$

Here the vector Π_k has value one at locations corresponding to the class k and zero for other entries. The class, d , which is associated with an observed vector, is then declared as the one that produces the smallest approximation error

$$d = \operatorname{argmin}_k r_k(\mathbf{y}). \quad (12)$$

The sparse representation-based classification method is summarized in Algorithm 1.

Algorithm 1. Sparse representation-based classification (SRC) algorithm

Input: $\mathbf{B} \in \mathbb{R}^{N \times (n \cdot L)}$, $\mathbf{y} \in \mathbb{R}^N$.

1. Solve the BP (8) or BPDN (9) problem.
2. Compute the residual using (11).
3. Identify \mathbf{y} using (12).

Output: Class label of \mathbf{y} .

For classification, it is important to be able to detect and then reject the test samples of poor quality. To decide whether a given test sample has good quality, one can use the notion of Sparsity Concentration Index (SCI) proposed in Wright et al. (2009). The SCI of a coefficient vector $\boldsymbol{\alpha} \in \mathbb{R}^{(L,n)}$ is defined as

$$\text{SCI}(\boldsymbol{\alpha}) = \frac{\frac{L \cdot \max_i \|\Pi_i(\boldsymbol{\alpha})\|_1}{\|\boldsymbol{\alpha}\|_1} - 1}{L - 1}. \quad (13)$$

SCI takes values between 0 and 1. SCI values close to 1 correspond to the case where the test image can be approximately represented by using only images from a single class. The test vector has enough discriminating features of its class, so has high quality. If $\text{SCI} = 0$ then the coefficients are spread evenly across all classes. So the test vector is not similar to any of the classes and has of poor quality. A threshold can be chosen to reject the images with poor quality. For instance, a test image can be rejected if $\text{SCI}(\hat{\boldsymbol{\alpha}}) < \lambda$ and otherwise accepted as valid, where λ is some chosen threshold between 0 and 1.

Table 1

Recognition rates (in %) of SRC algorithm (Wright et al., 2009) on the extended Yale B database

Dimension	30	56	120	504
Eigen	86.5	91.63	93.95	96.77
Laplacian	87.49	91.72	93.95	96.52
Random	82.60	91.47	95.53	98.09
Downsample	74.57	86.16	92.13	97.10
Fisher	86.91	—	—	—

Table 2

Recognition results with partial facial features (Wright et al., 2009)

	Right eye	Nose	Mouth
Dimension	5040	4270	12,936
SRC	93.7%	87.3%	98.3%
NN	68.8%	49.2%	72.7%
NS	78.6%	83.7%	94.4%
SVM	85.8%	70.8%	95.3%

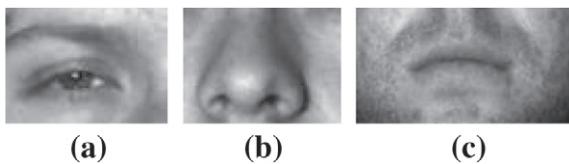


Fig. 2. Examples of partial facial features. (a) Eye, (b) nose, and (c) mouth.

2.2. Robust biometrics recognition via sparse representation

To illustrate the effectiveness of the SRC algorithm for face and iris biometrics, we highlight some of the results presented in Wright et al., (2009) and Pillai et al., (2011). The recognition rates achieved by the SRC method for face recognition with different features and dimensions are summarized in Table 1 on the extended Yale B Dataset (Georghiades et al., 2001). As it can be seen from Table 1 the SRC method achieves the best recognition rate of 98.09% with randomfaces of dimension 504.

Partial face features have been very popular in recovering the identity of human face (Sinha et al., 2006; Wright et al., 2009). The recognition results on partial facial features such as an eye, nose, and mouth are summarized in Table 2 on the same dataset. The SRC algorithm achieves the best recognition performance of 93.7%, 87.3%, and 98.3% on eye, nose, and mouth features, respectively, and it outperforms the other competitive methods such as Nearest Neighbor (NN), Nearest Subspace (NS), and Support Vector Machines (SVM). These results show that SRC can provide good recognition performance even in the case when partial face features are provided (see Fig. 2).

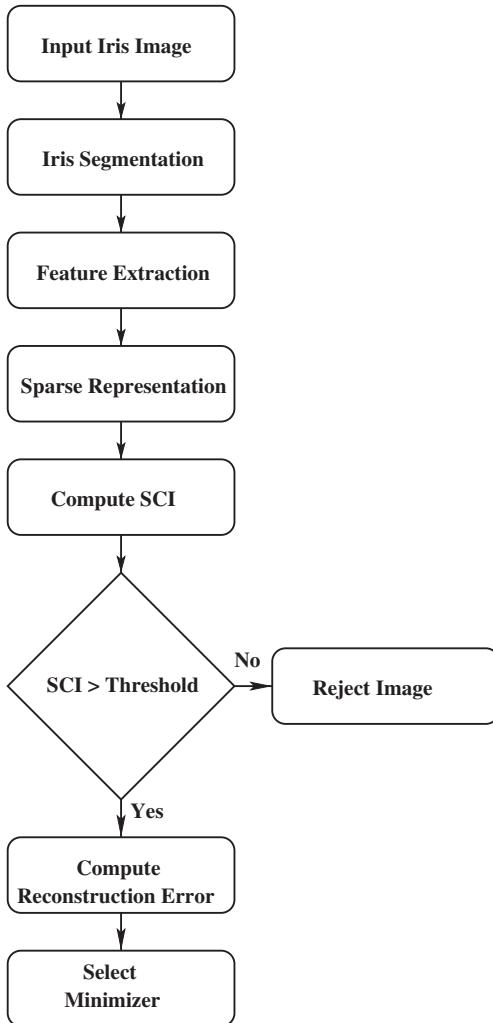


Fig. 3. Block diagram of the method proposed in Pillai et al. (2011) for the selection and recognition of iris images.

One of the main difficulties in iris biometric is that iris images acquired from a partially cooperating subject often suffer from blur, occlusion due to eyelids, and specular reflections. As a result, the performance of existing iris recognition systems degrade significantly on these images. Hence, it is essential to select good images before they are input to the recognition algorithm. To this end, one such algorithm based on SR for iris biometric was proposed in Pillai et al. (2011) that can select and recognize iris images in a single step. The block diagram of the method based on SR for iris recognition is shown in Fig. 3.

In Fig. 4, we display the iris images having the least SCI value for the blur, occlusion, and segmentation error experiments performed on the real iris images

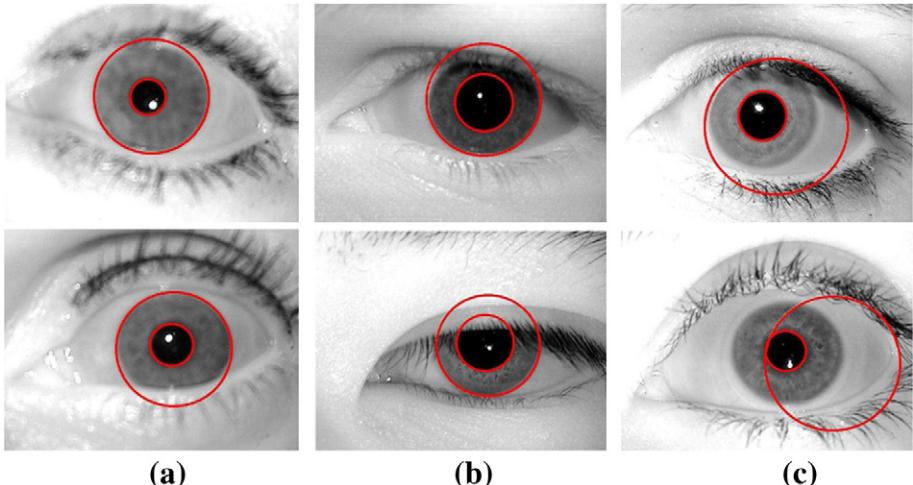


Fig. 4. Iris images with low SCI values in the ND dataset. Note that the images in (a), (b), and (c) suffer from high amounts of blur, occlusion, and segmentation errors, respectively.

Table 3
Recognition rate on ND dataset (Pillai et al., 2011)

Image quality	NN	Masek's implementation	SRC
Good	98.33	97.5	99.17
Blurred	95.42	96.01	96.28
Occluded	85.03	89.54	90.30
Seg. error	78.57	82.09	91.36

in the University of Notre Dame ND dataset (Bowyer and Flynn, 2004). As it can be observed, the low SCI images suffer from high amounts of distortion. The recognition performance of the SR-based method for iris biometric (Pillai et al., 2011) is summarized in Table 3. As it can be seen from the table SRC provides the best recognition performance over that of NN and Libor Masek's iris identification source code (Masek and Kovesi, 2003).

3. Dictionary learning

Instead of using a pre-determined dictionary \mathbf{B} , as in (1), one can directly learn it from the data (Olshausen and Field, 1996). In this section, we will highlight some of the methods for learning dictionaries and present their applications in object representation and classification (Rubinstein et al., 2010; Wright et al., 2010; Patel and Chellappa, 2010; Chen et al., 2012; Vishal et al., 2012; Shekhar et al., 2011; Shrivastava et al., 2012).

3.1. Dictionary learning algorithms

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) ([Engan et al., 1999](#)) and the KSVD algorithm ([Aharon et al., 2006](#)). Given a set of examples $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$, the goal of the KSVD and MOD algorithms is to find a dictionary \mathbf{B} and a sparse matrix Γ that minimize the following representation error

$$(\widehat{\mathbf{B}}, \widehat{\Gamma}) = \underset{\mathbf{B}, \Gamma}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{B}\Gamma\|_F^2 \quad \text{subject to } \|\boldsymbol{\gamma}_i\|_0 \leq T_0 \forall i,$$

where $\boldsymbol{\gamma}_i$ represent the columns of Γ 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{B} with ℓ_2 normalized columns is initialized. Then, the main iteration is composed of the following two stages:

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

$$i = 1, \dots, n, \quad \underset{\boldsymbol{\gamma}_i}{\min} \|\mathbf{x}_i - \mathbf{B}\boldsymbol{\gamma}_i\|_2^2 \text{ s.t. } \|\boldsymbol{\gamma}_i\|_0 \leq T_0.$$

Approximate solutions of the above problem can be obtained by using any sparse coding algorithms such Orthogonal Matching Pursuit (OMP) ([Pati et al., 1993; Tropp and Gilbert, 2006](#)) and BP ([Chen et al., 1998](#)).

- *Dictionary update*: This is where both MOD and KSVD algorithms differ. The MOD algorithm updates all the atoms simultaneously by solving the following optimization problem

$$\underset{\mathbf{B}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{B}\Gamma\|_F^2. \tag{14}$$

whose solution is given by

$$\mathbf{B} = \mathbf{X}\Gamma^T \left(\Gamma\Gamma^T \right)^{-1}.$$

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 as discussed in [Aharon et al. \(2006\)](#).

In the case of KSVD, the dictionary update is performed atom-by-atom in an efficient way rather than using a matrix inversion. The term to be minimized in Eq. (14) can be rewritten as:

$$\|\mathbf{X} - \mathbf{B}\Gamma\|_F^2 = \left\| \mathbf{X} - \sum_j \mathbf{b}_j \boldsymbol{\gamma}_j^T \right\|_2^2 = \left\| \left(\mathbf{X} - \sum_{j \neq j_0} \mathbf{b}_j \boldsymbol{\gamma}_j^T \right) - \mathbf{b}_{j_0} \boldsymbol{\gamma}_{j_0}^T \right\|_2^2,$$

where $\boldsymbol{\gamma}_j^T$ represents the j th row of Γ . Since we want to update \mathbf{b}_{j_0} and $\boldsymbol{\gamma}_{j_0}^T$, the first term in the above equation

$$\mathbf{E}_{j_0} = \mathbf{X} - \sum_{j \neq j_0} \mathbf{b}_j \boldsymbol{\gamma}_j^T$$

can be precomputed. The optimal \mathbf{b}_{j_0} and $\boldsymbol{\gamma}_{j_0}^T$ are found by an SVD decomposition. In particular, while fixing the cardinalities of all representations, a subset of the columns of \mathbf{E}_{j_0} is taken into consideration. This way of updating leads to a substantial speedup in the convergence of the training algorithm compared to the MOD method.

Both the MOD and the KSVD dictionary learning algorithms are described in detail in Algorithm 2

Algorithm 2. The MOD and KSVD dictionary learning algorithms

Objective: Find the best dictionary to represent the samples $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ as sparse compositions, by solving the following optimization problem:

$$\underset{\mathbf{B}, \boldsymbol{\Gamma}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{B}\boldsymbol{\Gamma}\|_F^2 \quad \text{subject to } \forall i \|\boldsymbol{\gamma}_i\|_0 \leq T_0.$$

Input: Initial dictionary $\mathbf{B}^{(0)} \in \mathbb{R}^{N \times P}$, with normalized columns, signal matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ and sparsity level T_0 .

1. *Sparse coding stage:*

Use any pursuit algorithm to approximate the solution of

$$\hat{\boldsymbol{\gamma}}_i = \underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \|\mathbf{x}_i - \mathbf{B}\boldsymbol{\gamma}\|_2^2 \quad \text{subject to } \|\boldsymbol{\gamma}\|_0 \leq T_0$$

obtaining sparse representation vector $\hat{\boldsymbol{\gamma}}_i$ for $1 \leq i \leq n$. These form the matrix $\boldsymbol{\Gamma}$.

2. *Dictionary update stage:*

MOD: Update the dictionary by the formula

$$\mathbf{B} = \mathbf{X}\boldsymbol{\Gamma}^T \left(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^T\right)^{-1}.$$

KSVD: For each column $k = 1, \dots, P$ in $\mathbf{B}^{(J-1)}$ update by

– Define the group of examples that use this atom,

$$\omega_k = \left\{ i | 1 \leq i \leq P, \boldsymbol{\gamma}_k^T(i) \neq 0 \right\}.$$

– Compute the overall representation error matrix, \mathbf{E}_k , by

$$\mathbf{E}_k = \mathbf{X} - \sum_{j \neq k} \mathbf{b}_j \boldsymbol{\gamma}_j^T.$$

– Restrict \mathbf{E}_k by choosing only the columns corresponding to ω_k and obtain \mathbf{E}_k^R .

– Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U} \Delta \mathbf{V}^T$. Select the updated dictionary column $\hat{\mathbf{b}}_k$ to be the first column of \mathbf{U} . Update the coefficient vector $\boldsymbol{\gamma}_k^R$ to be the first column of \mathbf{V} multiplied by $\Delta(1, 1)$.

3. Set $J = J + 1$.

Output: Trained dictionary \mathbf{B} and sparse coefficient matrix $\boldsymbol{\Gamma}$.

3.2. Discriminative dictionary learning

Dictionaries can be trained for both reconstruction and discrimination applications. In the late 1990s, Etemand and Chellappa proposed a Linear Discriminant Analysis (LDA)-based basis selection and feature extraction algorithm for classification using wavelet packets (Etemand and Chellappa, 1998). Recently, similar algorithms for simultaneous sparse signal representation and discrimination have also been proposed in Feng et al. (2011), Rodriguez and Sapiro (2007), Kokopoulou and Frossard (2008), Huang and Aviyente (2007), and Patel et al. (2010). The basic idea in learning a discriminative dictionary is to add an LDA type of discrimination on the sparse coefficients which essentially enforces separability among dictionary atoms of different classes. Some of the other methods for learning discriminative dictionaries include Mairal et al. (2012, 2008a,b), Zhang and Li (2010), Jiang et al. (2011), and Feng et al. (2011). Additional techniques may be found within these references.

In particular, a dictionary learning method based on information maximization principle was proposed in Qiu et al. (2011) for action recognition. The objective function in Qiu et al. (2011) maximizes the mutual information between what has been learned and what remains to be learned in terms of appearance information and class distribution for each dictionary item. A Gaussian Process (GP) model is proposed for sparse representation to optimize the dictionary objective function. The sparse coding property allows a kernel with a compact support in GP to realize a very efficient dictionary learning process. Hence, an action video can be described by a set of compact and discriminative action attributes.

Given the initial dictionary \mathbf{B}^o , the objective is to compress it into a dictionary \mathbf{B}^* of size k , which encourages the signals from the same class to have very similar sparse representations. Let L denote the labels of M discrete values, $L \in [1, M]$. Given a set of dictionary atoms \mathbf{B}^* , define $P(L|\mathbf{B}^*) = \frac{1}{|\mathbf{B}^*|} \sum_{\mathbf{b}_i \in \mathbf{B}^*} P(L|\mathbf{b}_i)$. For simplicity, denote $P(L|\mathbf{b}^*)$ as $P(L_{\mathbf{b}^*})$, and $P(L|\mathbf{B}^*)$ as $P(L_{\mathbf{B}^*})$.

To enhance the discriminative power of the learned dictionary, the following objective function is considered

$$\underset{\mathbf{B}^*}{\operatorname{argmax}} I(\mathbf{B}^*; \mathbf{B}^o \setminus \mathbf{B}^*) + \lambda I(L_{\mathbf{B}^*}; L_{\mathbf{B}^o \setminus \mathbf{B}^*}), \quad (15)$$

where $\lambda \geq 0$ is the parameter to regularize the emphasis on appearance or label information and I denotes mutual information. One can approximate (15) as

$$\begin{aligned} & \underset{\mathbf{b}^* \in \mathbf{B}^o \setminus \mathbf{B}^*}{\operatorname{argmax}} [H(\mathbf{b}^*|\mathbf{B}^*) - H(\mathbf{b}^*|\bar{\mathbf{B}}^*)] \\ & + \lambda [H(L_{\mathbf{b}^*}|L_{\mathbf{B}^*}) - H(L_{\mathbf{b}^*}|L_{\bar{\mathbf{B}}^*})], \end{aligned} \quad (16)$$

where H denotes entropy. One can easily notice that the above formulation also forces the classes associated with \mathbf{b}^* to be most different from classes already covered by the selected atoms \mathbf{B}^* ; and at the same time, the classes associated with \mathbf{b}^* are most representative among classes covered by the remaining atoms. Thus the learned dictionary is not only compact, but also covers all classes to maintain the discriminability.

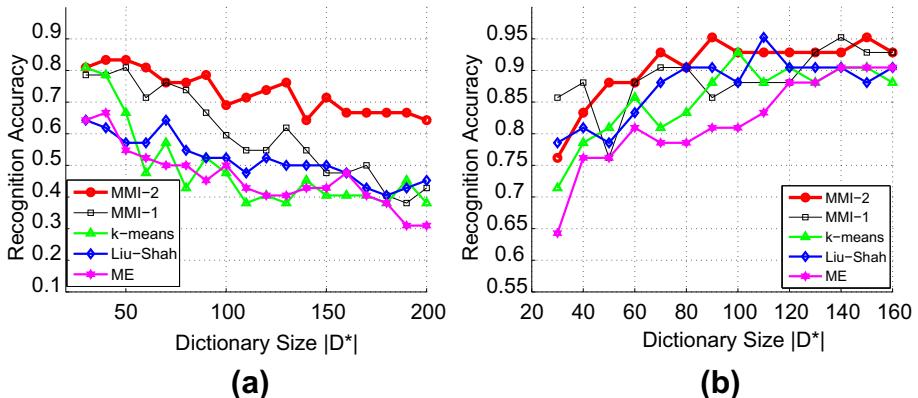


Fig. 5. Recognition accuracy on the Keck gesture dataset with different features and dictionary sizes (Shape and motion are global features. STIP is a local feature.) (Qiu et al., 2011). The recognition accuracy using initial dictionary D^0 : (a) 0.23, and (b) 0.42. In all cases, the MMI-2 (red line) outperforms the rest. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this book.)

In Fig. 5, we present the recognition accuracy on the Keck gesture dataset with different dictionary sizes and over different global and local features (Qiu et al., 2011). Leave-one-person-out setup is used. That is, sequences performed by a person are left out, and the average accuracy is reported. Initial dictionary size $|B^0|$ is chosen to be twice the dimension of the input signal and sparsity 10 is used in this set of experiments. As can be seen the mutual information-based method, denoted as MMI-2 outperforms the other methods.

Sparse representation over a dictionary with coherent atoms has the multiple representation problem. A compact dictionary consists of incoherent atoms, and encourages similar signals, which are more likely from the same class, to be consistently described by a similar set of atoms with similar coefficients (Qiu et al., 2011). A discriminative dictionary encourages signals from different classes to be described by either a different set of atoms, or the same set of atoms but with different coefficients (Rodriguez and Sapiro, 2007; Huang and Aviyente, 2007; Mairal et al., 2008a). Both aspects are critical for classification using sparse representation. The reconstructive requirement to a compact, and discriminative dictionary enhances the robustness of the discriminant sparse representation (Rodriguez and Sapiro, 2007). Hence, learning reconstructive, compact, and discriminative dictionaries is important for classification using sparse representation. Motivated by this observation, Qiu et al. (submitted for publication) proposed a general information theoretic approach to leaning dictionaries that are simultaneously reconstructive, discriminative, and compact.

Suppose that we are given a set of n signals (images) in an N -dim feature space $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n], \mathbf{x}_i \in \mathbb{R}^N$. Given that signals are from p distinct classes and N_c signals are from the c th class, $c \in \{1, \dots, p\}$, we denote $\Gamma = \{\Gamma_c\}_{c=1}^p$, where $\Gamma_c = [\gamma_1^c, \dots, \gamma_{N_c}^c]$ are signals in the c th class. Define $\mathbf{X} = \{\mathbf{X}_c\}_{c=1}^p$, where $\mathbf{X}_c = [\mathbf{y}_1^c, \dots, \mathbf{y}_{N_c}^c]$ is the sparse representation of \mathbf{X}_c .

Given \mathbf{X} and an initial dictionary \mathbf{B}^o with ℓ_2 normalized columns, a compact, reconstructive, and discriminative dictionary \mathbf{B}^* is learned via maximizing the mutual information between \mathbf{B}^* and the unselected atoms $\mathbf{B}^o \setminus \mathbf{B}^*$ in \mathbf{B}^o , between the sparse codes $\Gamma_{\mathbf{B}^*}$ associated with \mathbf{B}^* and the signal class labels C , and finally between the signals \mathbf{X} and \mathbf{B}^* , i.e.,

$$\underset{\mathbf{B}}{\operatorname{argmax}} \lambda_1 I(\mathbf{B}; \mathbf{B}^o \setminus \mathbf{B}) + \lambda_2 I(\Gamma_{\mathbf{B}}; C) + \lambda_3 I(\mathbf{X}; \mathbf{B}), \quad (17)$$

where $\{\lambda_1, \lambda_2, \lambda_3\}$ are the parameters to balance the contributions from compactness, discriminability, and reconstruction terms, respectively.

A two-stage approach is adopted to satisfy (17). In the first stage, each term in (17) is maximized in a unified greedy manner and involves a closed-form evaluation, thus atoms can be greedily selected from the initial dictionary while satisfying (17). In the second stage, the selected dictionary atoms are updated using a simple gradient ascent method to further maximize

$$\lambda_2 I(\Gamma_{\mathbf{B}}; C) + \lambda_3 I(\mathbf{X}; \mathbf{B}).$$

To illustrate how the discriminability of dictionary atoms selected by the information theoretic dictionary section (ITDS) method can be further enhanced using the information theoretic dictionary update (ITDU) method, consider Fig. 6. The Extended YaleB face dataset (Georghiades et al., 2001) and the USPS handwritten digits dataset [1] are used for illustration. Sparsity 2 is adopted for visualization, as the non-zero sparse coefficients of each image can now be plotted as a 2-D point. In Fig. 6, with a common set of atoms shared over all classes, sparse

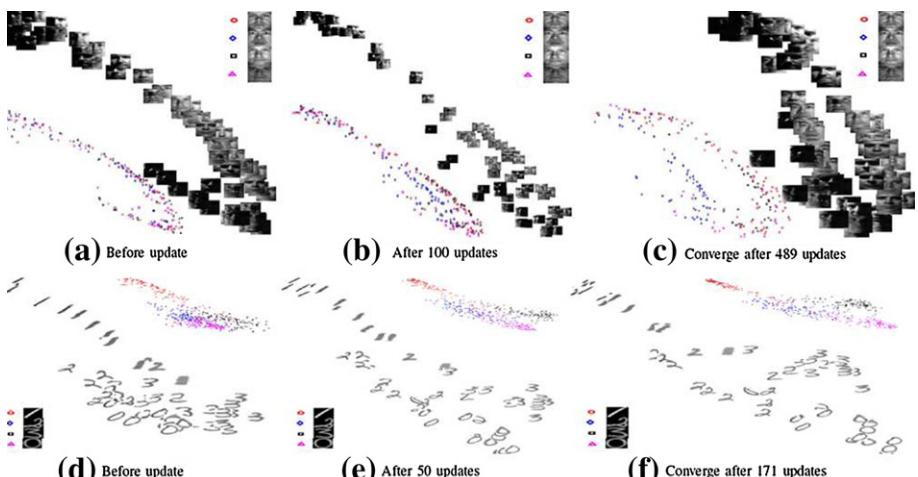


Fig. 6. Information-theoretic dictionary update with global atoms shared over classes. For a better visual representation, sparsity 2 is chosen and a randomly selected subset of all samples are shown. The recognition rates associated with (a), (b), and (c) are: 30.63%, 42.34%, and 51.35%. The recognition rates associated with (d), (e), and (f) are: 73.54%, 84.45%, and 87.75%. Note that the ITDU effectively enhances the discriminability of the set of common atoms (Qiu et al., submitted for publication).

coefficients of all samples become points in the same 2-D coordinate space. Different classes are represented by different colors. The original images are also shown and placed at the coordinates defined by their non-zero sparse coefficients. The atoms to be updated in Fig. 6a and d are selected using ITDS. It can be seen from Fig. 6 that the ITDU method makes sparse coefficients of different classes more discriminative, leading to significantly improved classification accuracy (Qiu et al., submitted for publication).

3.3. Non-linear kernel dictionary learning

Linear representations are almost always inadequate for representing non-linear data arising in many practical applications. For example, many types of descriptors in computer vision have intrinsic nonlinear similarity measure functions. The most popular ones include the spatial pyramid descriptor (Lazebnik et al., 2006) which uses a pyramid match kernel, and the region covariance descriptor (Tuzel et al., 2006) which uses a Riemannian metric as the similarity measure between two descriptors. Both of these distance measures are highly non-linear. Unfortunately, traditional sparse coding methods are based on linear models. This inevitably leads to poor performances for many datasets, e.g., object classification of Caltech-101 (Jiang et al., 2011) dataset, even when discriminant power is taken into account during the training. This has motivated researchers to study non-linear kernel sparse representation and dictionary learning for object recognition (Chen et al., 1233; Gao et al., 2010; Yuan and Yan, 2010; Zhang et al., 2012; Shekhar et al., 2012; Nguyen et al., 2012a).

To make the data in an input space separable, the data is implicitly mapped into a high-dimensional kernel feature space by using some nonlinear mapping associated with a kernel function. The kernel function, $\kappa : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$, is defined as the inner product

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle, \quad (18)$$

where $\phi : \mathbb{R}^N \rightarrow \mathcal{F} \subset \mathbb{R}^{\tilde{N}}$ is an implicit mapping projecting the vector \mathbf{x} into a higher dimensional space, \mathcal{F} . Some commonly used kernels include polynomial kernels

$$\kappa(\mathbf{x}, \mathbf{y}) = \langle (\mathbf{x}, \mathbf{y}) + c \rangle^d$$

and Gaussian kernels

$$\kappa(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{c}\right),$$

where c and d are the parameters. One can learn a non-linear dictionary \mathbf{B} in the feature space \mathcal{F} by solving the following optimization problem:

$$\underset{\mathbf{B}, \Gamma}{\operatorname{argmin}} \quad \|\Phi(\mathbf{X}) - \mathbf{B}\Gamma\|_F^2 \quad \text{s.t.} \quad \|\gamma_i\|_0 \leq T_0, \forall i, \quad (19)$$

where $\mathbf{B} \in \mathbb{R}^{\tilde{N} \times K}$ is the sought dictionary, $\mathbf{\Gamma} \in \mathbb{R}^{K \times n}$ is a matrix whose i th column is the sparse vector \mathbf{y}_i corresponding to the sample \mathbf{x}_i , with maximum of T_0 non-zero entries and with the abuse of notation we denote $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n)]$. It was shown in Nguyen et al. (2012a) that there exists an optimal solution \mathbf{B}^* to the problem (19) that has the following form:

$$\mathbf{B}^* = \Phi(\mathbf{X})\mathbf{A} \quad (20)$$

for some $\mathbf{A} \in \mathbb{R}^{n \times K}$. Moreover, this solution has the smallest Frobenius norm among all optimal solutions. As a result, one can seek an optimal dictionary through optimizing \mathbf{A} instead of \mathbf{B} . By substituting (20) into (19), the problem can be rewritten as follows:

$$\underset{\mathbf{A}, \mathbf{\Gamma}}{\operatorname{argmin}} \|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{A}\mathbf{\Gamma}\|_F^2 \text{ s.t. } \|\mathbf{y}_i\|_0 \leq T_0, \forall i. \quad (21)$$

In order to see the advantage of this formulation over the original one, we will examine the objective function. Through some manipulation, the cost function can be rewritten as:

$$\|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{A}\mathbf{\Gamma}\|_F^2 = \text{tr}((\mathbf{I} - \mathbf{A}\mathbf{\Gamma})^T \mathbb{K}(\mathbf{X}, \mathbf{X}) (\mathbf{I} - \mathbf{A}\mathbf{\Gamma})),$$

where $\mathbb{K}(\mathbf{X}, \mathbf{X})$ is a kernel matrix whose elements are computed from $\kappa(i, j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$. It is apparent that the objective function is feasible since it only involves a matrix of finite dimension $\mathbb{K} \in \mathbb{R}^{n \times n}$, instead of dealing with a possibly infinite dimensional dictionary. An important property of this formulation is that the computation of \mathbb{K} only requires dot products. Therefore, we are able to employ *Mercer* kernel functions to compute these dot products without carrying out the mapping Φ .

To solve the above optimization problem for learning non-linear dictionaries, variants of MOD and K-SVD algorithms in the feature space have been proposed Nguyen et al. (2012a,b). The procedure essentially involves two stages: sparse coding and dictionary update in the feature space. For sparse coding, one can adapt the non-linear version of orthogonal matching pursuit algorithm (Nguyen et al., 2012a). Once the sparse codes are found in the feature space, the dictionary atoms are updated in an efficient way (Nguyen et al., 2012a,b).

The optimization problem (21) is purely generative. It does not explicitly promote the discrimination which is important for many classification tasks. Using the kernel trick, when the data is transformed into a high-dimensional feature space, the data from different classes may still overlap. Hence, generative dictionaries may lead to poor performance in classification even when data is non-linearly mapped to a feature space. To overcome this, a method for designing non-linear dictionaries that are simultaneously generative and discriminative was proposed in Shrivastava et al. (2012).

Figure 7 presents an important comparison in terms of the discriminative power of learning a discriminative dictionary in the feature space where kernel LDA type of discriminative term has been included in the objective function. A scatter plot of the sparse coefficients obtained using different approaches shows that such a discriminative dictionary is able to learn the underlying non-linear sparsity of

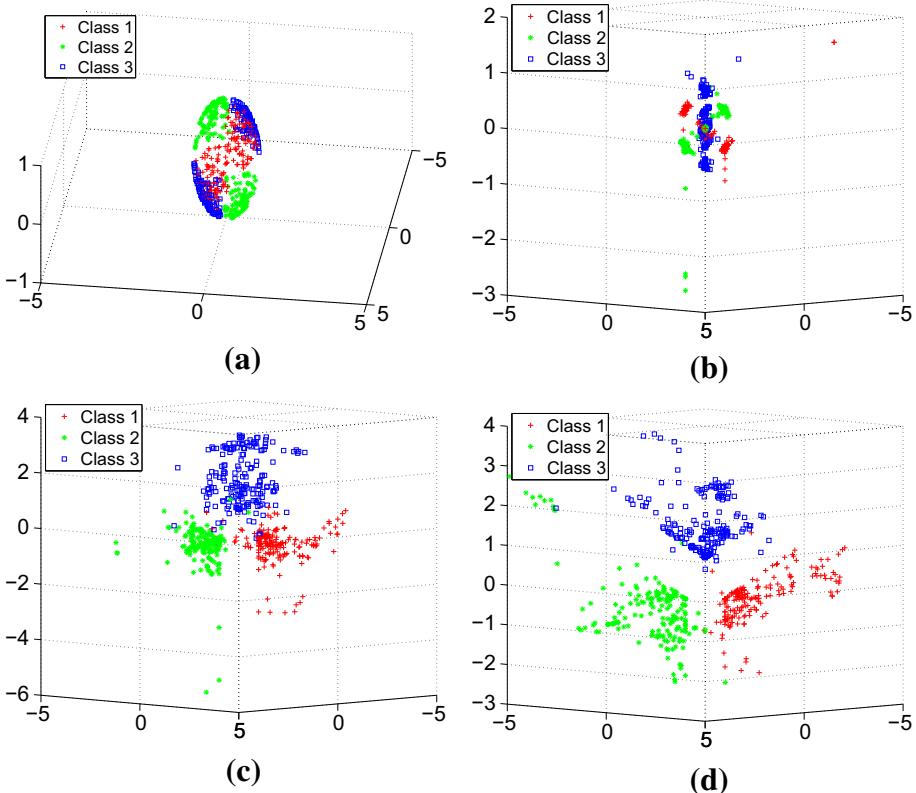


Fig. 7. A synthetic example showing the significance of learning a discriminative dictionary in feature space for classification. (a) Synthetic data which consists of linearly non-separable 3D points on a sphere. Different classes are represented by different colors. (b) Sparse coefficients from K-SVD projected onto learned SVM hyperplanes. (c) Sparse coefficients from a non-linear dictionary projected onto learned SVM hyperplanes. (d) Sparse coefficients from non-linear discriminative kernel dictionary projected onto learned SVM hyperplanes (Shrivastava et al., 2012). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this book.)

data as well as it provides more discriminative representation. See Shrivastava et al. (2012), and Nguyen et al. (2012a,b) for more details on the design of non-linear kernel dictionaries.

3.4. Joint dimensionality reduction and dictionary learning

Signals are usually assumed to lie on a low-dimensional manifold embedded in a high-dimensional space. Dealing with the high-dimension is not practical for both learning and inference tasks. As an example of the effect of dimension on learning, Stone, (1982) showed that, under certain regularity assumption including that samples are identically independent distributed, the optimal rate of convergence for non-parametric regression decreases exponentially with the dimension of the data. As the dimension increases, the Euclidean distances between feature vectors

become closer to each other making the inference task harder. This is known as the concentration phenomenon (Beyer et al., 1999). To address these issues, various linear and non-linear dimensionality reduction (DR) techniques have been developed (Lee and Verleysen, 2006). Some examples include PCA (Pearson, 1901), ISOMAP (Tenenbaum et al., 2000), LLE (Roweis and Saul, 2000), Laplacian Eigenmaps (Belkin et al., 2003), etc. In general, these techniques map data to a lower dimensional space such that non-informative or irrelevant information in the data is discarded.

As we saw in the previous sections, dictionary learning methods have been popular for representing and processing of signals. However, the current algorithms for finding a good dictionary have some drawbacks. The learning of \mathbf{B} is challenging due to the high-dimensional nature of the training data, as well as the lack of training samples. Therefore, DR seems to be a natural solution. Unfortunately, the current DR techniques are not designed to respect and promote underlying sparse structures of data. Therefore, they cannot help the process of learning the dictionary \mathbf{B} .

An interesting framework, called *sparse embedding* (SE), that brings the strength of both dimensionality reduction and sparse learning together was recently proposed in Nguyen et al. (2012c). In this framework, the dimension of signals is reduced in a way such that the sparse structures of signals are promoted. The algorithm simultaneously learns a dictionary in the reduced space, yet, allows the recovery of the dictionary in the original domain. This empowers the algorithm with two important advantages: (1) Ability to remove the distracting part of the signal that negatively interferes with the learning process and (2) learning in the reduced space with smaller computational complexity. In addition, the framework is able to handle sparsity in a non-linear model through the use of the Mercer kernels.

Different from the classical approaches, the algorithm embeds input signals into a low-dimensional space and simultaneously learns an optimized dictionary. Let \mathcal{M} denote the mapping that transforms input signals into the output space. In general, \mathcal{M} can be non-linear. However, for the simplicity of notations, we temporarily restrict our discussion to linear transformations. As a result, the mapping \mathcal{M} is characterized using a matrix $\mathbf{P} \in \mathbb{R}^{d \times N}$. One can learn the mapping together with the dictionary through minimizing some appropriate cost function \mathcal{C}_X :

$$\{\mathbf{P}^*, \mathbf{B}^*, \boldsymbol{\Gamma}^*\} = \underset{\mathbf{P}, \mathbf{B}, \boldsymbol{\Gamma}}{\operatorname{argmin}} \mathcal{C}_X(\mathbf{P}, \mathbf{B}, \boldsymbol{\Gamma}). \quad (22)$$

This cost function \mathcal{C}_X needs to have several desirable properties. First, it has to promote sparsity within the reduced space. At the same time, the transformation \mathbf{P} resulting from optimizing \mathcal{C}_X has to retain useful information present in the original signals. A second criterion is also needed in order to prevent the pathological case when everything is mapped into the origin, which obtains the sparsest solution but is obviously of no interest. Toward this end, the following optimization was proposed in Nguyen et al. (2012c):

$$\begin{aligned} \{\mathbf{P}^*, \mathbf{B}^*, \boldsymbol{\Gamma}^*\} &= \underset{\mathbf{P}, \mathbf{B}, \boldsymbol{\Gamma}}{\operatorname{argmin}} \left(\|\mathbf{P}\mathbf{X} - \mathbf{B}\boldsymbol{\Gamma}\|_F^2 + \lambda \|\mathbf{X} - \mathbf{P}^T \mathbf{P}\mathbf{X}\|_F^2 \right) \\ \text{subject to: } \mathbf{P}\mathbf{P}^T &= \mathbf{I} \quad \text{and } \|\boldsymbol{\gamma}_i\|_0 \leq T_0, \forall i, \end{aligned} \quad (23)$$

where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is the identity matrix, λ is a positive constant, and the dictionary is now in the reduced space, i.e., $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_K] \in \mathbb{R}^{d \times K}$. The first term of the cost function promotes sparsity of signals in the reduced space. The second term is the amount of energy discarded by the transformation \mathbf{P} , or the difference between low-dimensional approximations and the original signals. In fact, the second term is closely related to PCA as by removing the first term, it can be shown that the solution of \mathbf{P} coincides with the principal components of the largest eigenvalues, when the data are centered.

A simple two step procedure for finding both the projection mapping and learning linear as well as non-linear dictionaries was proposed in Nguyen et al. (2012c). It was shown that sparse embedding can capture the meaningful structure of data and can perform significantly better than many competitive algorithms on signal recovery and object classification tasks.

3.5. Unsupervised dictionary learning

Dictionary learning techniques for unsupervised clustering have also gained some traction in recent years. In Sprechmann and Sapiro (2010), a method for simultaneously learning a set of dictionaries that optimally represent each cluster is proposed. To improve the accuracy of sparse coding, this approach was later extended by adding a block incoherence term in their optimization problem (Ramirez et al., 2010). Additional sparsity motivated subspace clustering methods include Elhamifar and Vidal (2009), Rao et al., (2008), Soltanolkotab and Candès (2011).

In particular, scale and in-plane rotation invariant clustering approach, which extends the dictionary learning and sparse representation framework for clustering and retrieval of images was proposed in Chen et al. (2012). Figure 8 presents and overview this approach (Chen et al., 2012). Given a database of images $\{\mathbf{x}_j\}_{j=1}^N$ and the number of clusters K , the Radon transform (Helgason, 1980) is used to find scale and rotation invariant features. It then uses sparse representation methods to simultaneously cluster the data and learn dictionaries for each cluster. One of the main features of this method is that it is effective for both texture- and shape-based images. Various experiments in Chen et al. (2012) demonstrated the effectiveness of this approach in image retrieval experiments, where the significant improvements in performance are achieved.

3.6. Dictionary learning from partially labeled data

The performance of a supervised classification algorithm is often dependent on the quality and diversity of training images, which are mainly hand labeled. However, labeling images are expensive and time consuming due to the significant human effort involved. On the other hand, one can easily obtain large amounts of unlabeled images from public image datasets like Flickr or by querying image search engines like Bing. This has motivated researchers to develop semi-supervised algorithms, which utilize both labeled and unlabeled data for learning classifier models. Such methods have demonstrated improved performance when the amount of labeled data is limited. See Chapelle et al. (2006) for an excellent survey of recent efforts on semi-supervised learning.

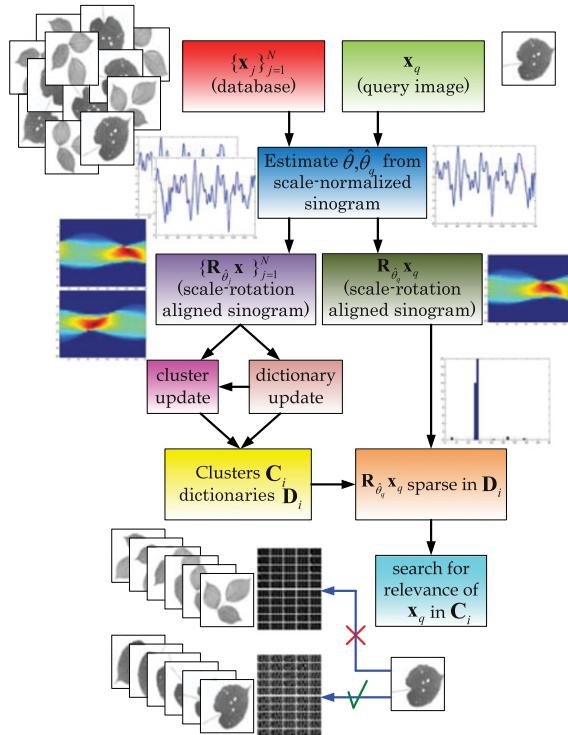


Fig. 8. Overview of simultaneous scale and in-plane rotation invariant clustering and dictionary learning method (Chen et al., 2012).

Two of the most popular methods for semi-supervised learning are Co-Training (Blum and Mitchell, 1998) and Semi-Supervised Support Vector Machines (S3VM) (Sindhwani and Keerthi, 2006). Co-Training assumes the presence of multiple views for each feature and uses the confident samples in one view to update the other. However, in applications such as image classification, one often has just a single feature vector and hence it is difficult to apply Co-Training. Semi-supervised support vector machines consider the labels of the unlabeled data as additional unknowns and jointly optimizes over the classifier parameters and the unknown labels in the SVM framework (Burges, 1998).

An interesting method to learn discriminative dictionaries for classification in a semi-supervised manner was recently proposed in Shrivastava et al. (2012). Figure 9 shows the block diagram of this method (Shrivastava et al., 2012) which uses both labeled and unlabeled data. While learning a dictionary, probability distribution is maintained over class labels for each unlabeled data. The discriminative part of the cost is made proportional to the confidence over the assigned label of the participating training sample. This makes the method robust to label assignment errors. See Shrivastava et al. (2012) for more details on the optimization of the partially labeled dictionary learning.

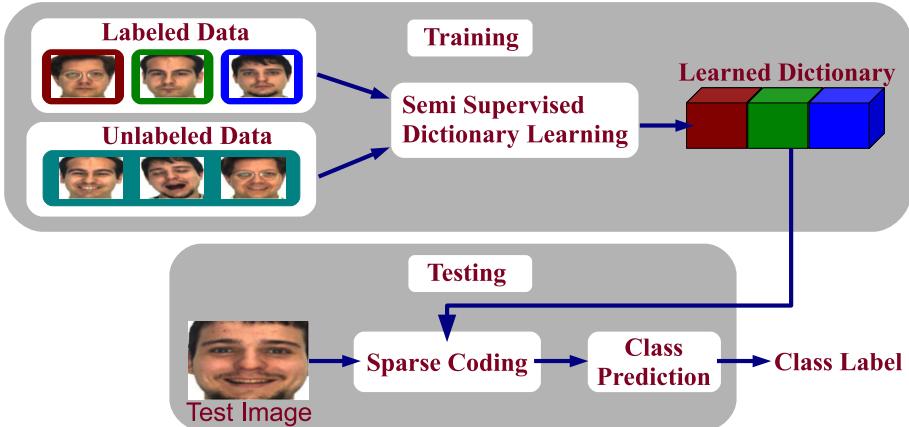


Fig. 9. Block diagram illustrating semi-supervised dictionary learning (Shrivastava et al., 2012).

4. Concluding remarks

In this chapter, we reviewed the role of sparse representation and dictionary learning for object recognition in supervised, semi-supervised, unsupervised, and weakly supervised setting. Furthermore, through the use of Mercer kernels, we showed how sparse representation and dictionary learning methods can be made non-linear. Even though, the main emphasis was given to object recognition, these methods can offer compelling solutions to other computer vision and machine learning problems such as matrix factorization, tracking, object detection, object recognition from video (Chen et al., 2012) and domain adaptation (Qiu et al., 2012).

References

- Aharon, M., Elad, M., Bruckstein, A.M., 2006. The k-svd: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.* 54 (11), 4311–4322.
- Belkin, Mikhail, Niyogi, Partha, 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.* 15 (6), 1373–1396.
- Beyer Kevin, Goldstein Jonathan, Ramakrishnan Raghu, Shaft Uri, 1999. When is nearest neighbor meaningful? In: ICDT: 7th International Conference on Database Theory.
- Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training. In: ACM Conference on Computational Learning Theory.
- Bowyer, K.W., Flynn, P.J., 2004. The nd-iris-0405 iris image dataset. Notre Dame CVRL Technical Report.
- Bruckstein, Alfred M., Donoho, David L., Elad, Michael, 2009. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Rev.* 51 (1), 34–81.
- Burges, C.J.C., 1998. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Disc.* 2, 121–167.
- Chapelle, O., Schölkopf, B., Zien, A., 2006. Semi-supervised Learning. Adaptive Computation and Machine Learning. MIT Press.
- Chen, Yi, Nasrabadi, N.M., Tran, T.D., 2011. Hyperspectral image classification via kernel sparse representation. In: IEEE International Conference on Image Processing, September, pp. 1233–1236.

- Chen, S., Donoho, D., Saunders, M., 1998. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comp.* 20 (1), 33–61.
- Chen, Y.-C., Patel, V.M., Phillips, P.J., Chellappa, R., 2012. Dictionary-based face recognition from video. In: European Conference on Computer Vision.
- Chen, Y.-C., Sastry, C.S., Patel, V.M., Phillips, P.J., Chellappa, R., 2012. Rotation invariant simultaneous clustering and dictionary learning. In: IEEE International Conference on Acoustics, Speech and Signal Processing.
- Donoho, David L., Elad, Michael, 2003. Optimally sparse representation in general (non-orthogonal) dictionaries via ℓ_1 minimization. *Proc. Natl Acad. Sci.* 100, 2197–2202.
- Elad, M., Figueiredo, M.A.T., Ma, Y., 2010. On the role of sparse and redundant representations in image processing. *Proc. IEEE* 98 (6), 972–982.
- Elhamifar, E., Vidal, R., 2009. Sparse subspace clustering. In: Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR), pp. 2790–2797.
- Engan, K., Aase, S.O., Husoy, J.H., 1999. Method of optimal directions for frame design. *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* 5, 2443–2446.
- Etemand, K., Chellappa, R., 1998. Separability-based multiscale basis selection and feature extraction for signal and image classification. *IEEE Trans. Image Process.* 7 (10), 1453–1465.
- Feng, X., Yang, M., Zhang, L., Zhang, D., 2011. Fisher discrimination dictionary learning for sparse representation. In: International Conference on Computer Vision.
- Gao, S., Tsang, I.W., Chia, L.-T., 2010. Kernel sparse representation for image classification and face recognition. In: European Conference on Computer Vision, vol. 6314.
- Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J., 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (6), 643–660.
- Gribonval, R., Nielsen, M., 2003. Sparse representations in unions of bases. *IEEE Trans. Inform. Theory* 49 (12), 3320–3325.
- Helgason, S., 1980. The Radon Transform. Birkhauser, Boston.
- Huang, K., Aviyente, S., 2007. Sparse representation for signal classification. In: Neural Information Processing Systems, vol. 19, pp. 609–616.
- Jiang, Zhuolin, Lin, Zhe, Davis, Larry S., 2011. Learning a discriminative dictionary for sparse coding via label consistent k-svd, In: Computer Vision and Pattern Recognition.
- Jiang, Zhuolin, Lin, Zhe, Davis, L.S., 2011. Learning a discriminative dictionary for sparse coding via label consistent k-svd. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2011, pp. 1697–1704.
- Kokiopoulou, E., Frossard, P., 2008. Semantic coding by supervised dimensionality reduction. *IEEE Trans. Multimedia* 10 (5), 806–818.
- Lazebnik, S., Schmid, C., Ponce, J., 2006. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2006, vol. 2, pp. 2169–2178.
- Lee, J.A., Verleysen, M., 2006. Nonlinear dimensionality reduction. In: Information Science and Statistics, Springer.
- Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A., 2008a. Discriminative learned dictionaries for local image analysis. In: Proceedings of the Conference on Computer Vision and, Pattern Recognition.
- Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A., 2008b. Supervised dictionary learning. *Adv. Neural Inform. Process. Syst.*
- Mairal, J., Bach, F., Ponce, J., 2012. Task-driven dictionary learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 34 (4), 791–804.
- Masek, L., Kovesi, P., 2003. Matlab Source Code for a Biometric Identification System Based on Iris Patterns. The School of Computer Science and Software Engineering, The University of Western Australia.
- Nguyen, H.V., Patel, V.M., Nasrabadi, N.M., Chellappa, R., 2012a. Design of non-linear kernel dictionaries for object recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*
- Nguyen, H.V., Patel, V.M., Nasrabadi, N.M., Chellappa, R., 2012b. Kernel dictionary learning. In: IEEE International Conference on Acoustics, Speech and Signal Processing.
- Nguyen, H.V., Patel, V.M., Nasrabadi, N.M., Chellappa, R., 2012c. Sparse embedding: a framework for sparsity promoting dimensionality reduction. In: European Conference on Computer Vision.

- Olshausen, B.A., Field, D.J., 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381 (6583), 607–609.
- Patel, V.M., Chellappa, R., 2010. Sparse representations, compressive sensing and dictionaries for pattern recognition. In: Asian Conference on Pattern Recognition.
- Patel, V.M., Easley, G.R., Healy, D.M., 2010. Automatic target recognition based on simultaneous sparse representation. In: IEEE International Conference on Image Processing.
- Pati, Y.C., Rezaifar, R., Krishnaprasad, P.S., 1993. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: Proceedings of the 27th Annual Asilomar Conference Signals, Systems, and Computers, November.
- Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. *Lond. Edinburgh Dublin Phil. Mag. J. Sci.* 2, 559–572.
- Pillai, J., Patel, V.M., Chellappa, R., 2009. Sparsity inspired selection and recognition of iris images. In: Third IEEE International Conference on Biometrics: Theory, Applications and Systems.
- Pillai, Jaishankar K., Patel, Vishal M., Chellappa, Rama, Ratha, Nalini K., 2011. Secure and robust iris recognition using random projections and sparse representations. *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (9), 1877–1893.
- Qiu, Qiang, Jiang, Zhuolin, Rama Chellappa, 2011. Sparse dictionary-based representation and recognition of action attributes. In: International Conference on Computer Vision.
- Qiu, Q., Patel, V.M., Turaga, P., Chellappa, R., 2012. Domain adaptive dictionary learning. In: European Conference on Computer Vision.
- Qiu, Q., Patel, V.M., Chellappa, R., submitted for publication. Information-theoretic dictionary learning for image classification. *IEEE Trans. Pattern Anal. Mach. Intell.*
- Ramirez, I., Sprechmann, P., Sapiro, G., 2010. Classification and clustering via dictionary learning with structured incoherence and shared features. In: Proceeding of IEEE Conference Computer Vision and Pattern Recognition (CVPR), pp. 3501–3508.
- Rao, S.R., Tron, R., Vidal, R., Ma, Y., 2008. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In: Proceeding of IEEE Conference Computer Vision and Pattern Recognition (CVPR), June, pp. 1–8.
- Rodriguez, F., Sapiro, G., 2007. Sparse Representations for Image Classification: Learning Discriminative and Reconstructive Non-Parametric Dictionaries. University of Minnesota. Tech. Report, December.
- Roweis, S.T., Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290 (5500), 2323–2326.
- Rubinstein, R., Bruckstein, A.M., Elad, M., 2010. Dictionaries for sparse representation modeling. *Proc. IEEE* 98 (6), 1045–1057.
- Shekhar, S., Patel, V.M., Nasrabadi, N.M., Chellappa, R., 2012. Joint sparsity-based robust multimodal biometrics recognition. European Conference on Computer Vision (ECCV) Workshop on Information Fusion in Computer Vision for Concept Recognition (IFCVCR), Florence, Italy.
- Shekhar, S., Patel, V.M., Chellappa, R., 2011. Synthesis-based recognition of low resolution faces. In: International Joint Conference on Biometrics, October, pp. 1–6.
- Shrivastava, A., Pillai, J.K., Patel, V.M., Chellappa, R., 2012. Learning discriminative dictionaries with partially labeled data. In: IEEE International Conference on Image Processing, pp. 1–14.
- Shrivastava, A., Nguyen, H.V., Patel, V.M., Chellappa, R., 2012. Design of non-linear discriminative dictionaries for image classification. In: Asian Conference on Computer Vision, Daejeon, Korea.
- Sindhwan V., Keerthi, S.S., 2006. Large scale semi-supervised linear svms. In: ACM Special Interest Group on Information Retrieval.
- Sinha, P., Balas, B., Ostrovsky, Y., Russell, R., 2006. Face recognition by humans: nineteen results all computer vision researchers should know about. *Proc. IEEE* 94 (11), 1948–1962.
- Soltanolkotab Mahdi, Candnes Emmanuel J., 2011. A geometric analysis of subspace clustering with outliers. Preprint.
- Sprechmann, P., Sapiro, G., 2010. Dictionary learning and sparse coding for unsupervised clustering. In: Proceeding of IEEE Conference International Conference on Acoustics, Speech, and Signal Processing (ICASSP), March, pp. 2042–2045.
- Stone, C.J., 1982. Optimal global rates of convergence for nonparametric regression. *Ann. Stat.* 10, 1040–1053.

- Tenenbaum, J.B., de Silva, V., Langford, J.C., 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290 (5500), 2319–2323.
- Tropp, J.A., Gilbert, A.C., 2006. Signal recovery from partial information via orthogonal matching pursuit. *IEEE Trans. Inform. Theory* 53 (12), 4655–4666.
- Tuzel, O., Porikli, F.M., Meer, P., 2006. Region covariance: a fast descriptor for detection and classification. In: European Conference on Computer Vision, pp. II:589–II:600.
- Usps handwritten digit database. In: <<http://www-i6.informatik.rwth-aachen.de/keysers/usps.html>>.
- Patel, Vishal M., Wu, Tao, Biwas, Soma, Jonathon Phillips, P., Chellappa, Rama, 2012. Dictionary-based face recognition under variable lighting and pose. *IEEE Trans. Inform. Forensics Secur.* 7 (3), 954–965.
- Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y., 2009. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 31 (2), 210–227.
- Wright, J., Ma, Y., Mairal, J., Sapiro, G., Huang, T.S., Shuicheng, Yan, 2010. Sparse representation for computer vision and pattern recognition. *Proc. IEEE* 98 (6), 1031–1044.
- Yuan, X.-T., Yan, S., 2010. Visual classification with multi-task joint sparse representation. In: Computer Vision and Pattern Recognition.
- Zhang, Q., Li, B., 2010. Discriminative k-svd for dictionary learning in face recognition. In: Computer Vision and Pattern Recognition.
- Zhang Li, Wei-Da Zhou, Pei-Chann Chang, Jing Liu, Zhe Yan, Ting Wang, Fan-Zhang Li. 2012. Kernel sparse representation-based classifier. *IEEE Trans. Signal Process.* 60 (4), 1684–1695.

This page is intentionally left blank

Conditional Random Fields for Scene Labeling

Ifeoma Nwogu and Venu Govindaraju

*Department of Computer Science and Engineering, University at Buffalo,
SUNY, Buffalo, NY 14260, USA*

Abstract

High-level, or holistic, scene understanding involves reasoning about objects, regions, the 3D relationships between them, etc. Scene labeling underlies many of these problems in computer vision. Reasoning about scene images requires the decomposition into semantically meaningful regions over which a graphical model can be imposed. Typically, representational models, learned from data, are defined in terms of a unified energy function over the appearance and structure of the scene-under-investigation. In this chapter, we explore energy functions defined within the context of conditional random fields (CRF) and examine in detail, one learning and inference technique that can be used to reason about the scene. We specifically review methods involving semantic categorization (such as grass, sky, foreground, etc.) and geometric categorization (typically the vertical plane, the ground plane, and the sky plane). CRFs are an effective tool for partitioning images into their constituent semantic or geometric level regions and assigning the appropriate class labels to each region. We also present specific algorithms from the literature that have successfully used CRFs for labeling scene images.

Keywords: conditional random fields, scene parsing, classification, probabilistic inference, learning graphical models

1. Introduction

Scene labeling underlies many problems in computer vision such as scene parsing, depth estimation from monocular images, object detection and recognition, etc. If we consider the scene in the image in Fig. 1, understanding this scene requires knowledge about the content, the setting, lighting conditions, and situation of the image. For example, we could recover much useful information such as the presence of people, buildings, a train, the color of the sky, etc., in the image. The goal in scene labeling is to recover the different regions of a scene along with their relationships to each



Fig. 1. A street scene in Hong Kong. Understanding such a scene involves estimating its different region content, its spatial layout, the overall scene setting, the situation of the scene, etc.

other. Having such a representation could then allow different objects to be physically “placed” within the correct frame and permit reasoning between the different objects and their 3D environment. Different problems in scene understanding that can be viewed as scene labeling problems include: (i) scene parsing which is the classical problem of partitioning the image into its component regions; (ii) recovering the surface layout in an image, which involves labeling the image into coarse geometric classes; (iii) background/foreground detection which involves labeling the image into two classes. Several forms of object detectors can also be viewed as scene labeling problems where the object-under-investigation is labeled with one class and everything else with another class.

In the last few years, the frameworks for scene labeling typically require a representation above the level of pixels that can be endowed with high-level attributes such as class of object/region, its orientation, and (rough 3D) location within the scene. Hence, such methods involve region-based models which combine features based on appearance and scene geometry, to automatically decompose a scene into semantically meaningful regions. These models define a unified energy function over scene appearance and structure in terms, where the energy function can be learned from data and efficient inference techniques are applied. These techniques begin with multiple over-segmentations of the image and propose moves in the energy-space to achieve multi-class image labeling.

Many scene labeling approaches in the field of computer vision implement probabilistic graphical models (PGM). Traditionally, PGMs have been used to represent the joint probability distribution $p(y, x)$ where y represents the attributes of the entities we wish to predict, and x , the input variables, represents the observed data from the world. But as well known, the modeling of joint distributions can be very challenging especially when modeling rich features are found in many scene understanding tasks. This is because the modeling of the joint probability distribution requires the modeling of the prior distribution over the data, $p(x)$. But the distribution $p(x)$ often involves complex dependencies and attempting to model these complexities can lead to intractable models. Meanwhile, ignoring them can lead to significantly reduced performance. But directly modeling

the conditional distribution $p(\mathbf{y}|\mathbf{x})$ alleviates this problem and is sufficient for performing classification problems, as found in scene labeling tasks. The process of modeling the conditional distribution does *not* involve the dependencies in \mathbf{x} , hence the resulting conditional model can have much simpler structure than its joint equivalent. It is also easier to learn and can yield more accurate results than the joint model. Hence, CRFs have played a very significant and successful role in the last few years, in advancing many computer vision tasks involving images and videos.

This chapter describes CRFs and its application for scene labeling in detail, and we do not assume extensive previous knowledge of graphical modeling. We begin by providing a general overview of the CRF structure, independent of its application to scene understanding (Section 2). In Section 3, we present general scene-specific implementations of CRFs, describing the standard features extracted along with the standard learning and inference procedures linking these general implementation details with the technical CRF structure details from Section 4. We also discuss specific implementation algorithms in the context of the general framework. Lastly in Section 5, we present a discussion of CRF in the context of scene labeling and outline a few open areas for future work.

2. Overview of CRF

Graphical modeling has proven useful as a framework for representation and inference in multi-variate probability distributions. Distributions over many variables can be expensive to represent. Even when the dimensionality of the data is small such as in binary data, the table of joint probabilities for n such variables will require storage of $O(2^n)$ floating numbers. Hence, the graphical modeling framework introduces a concept where a distribution over such input data can be represented as a product of local functions that each depends on a much smaller subset of variables. This process of breaking largely interconnects structures into products of local functions is called *factorization* and much of the power of the graphical modeling framework stems from this relationship between factorization, conditional independence, and graph structure. Conditional independence is useful for designing models and factorization is useful for designing inference algorithms. A more detailed perspective on graphical modeling and approximate inference methods can be found in the tutorial by Sutton and McCallum (2010) and the textbook by Koller and Friedman (2009).

2.1. Undirected graphical models

A CRF may be viewed as an undirected graphical model, or Markov random field (Clifford, 1990), globally conditioned on \mathbf{X} , the random input variables representing observed data. Formally, we define $G = (V, E)$ to represent an undirected graph such that there is a node $s \in V$ corresponding to each of the random variables, and takes outcomes from a set \mathcal{V} , which can be either continuous or discrete. Since we are primarily concerned with scene labeling as a classification task, we will only consider the discrete case in the article. For every variable $s \in V$, let $\mathcal{N}(s)$ represent the neighbors of s . We can therefore say that a distribution p is *Markov* with respect

to G if it meets the local Markov property: for any two variables $s, t \in V$, the variable s is independent of t conditioned on its neighbors $\mathcal{N}(s)$, implying that the neighbors of s contain all of the information necessary to predict its value and s is ignorant of all else in the network, given its neighbors.

Given a variable $s \in X$, the notation x_s denotes the value assigned to s by \mathbf{x} , where the vector \mathbf{x} denotes an arbitrary assignment to X . Similarly, the value assigned to a subset $a \subset X$ is denoted by \mathbf{x}_a . For marginalization, for a fixed variable assignment y_s , the summation $\sum_{\mathbf{y} \setminus y_s}$ indicates a summation over all possible assignments \mathbf{y} whose value for variable s is equal to y_s .

An undirected graphical model can therefore be defined as the family of probability distributions that factorize according to a given collection of subsets $F = a \subset V$. The set of all such distributions can be written in the form:

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_{a \subset F} \Psi_a(\mathbf{x}_a, \mathbf{y}_a). \quad (1)$$

$F = \{\Psi_a\}$ is known as the *local functions, factors, or compatibility functions*. Since all the variables in this distribution are considered random, the term *random field* is used to refer to a *particular distribution* among the set of all such distributions defined by an undirected model. The constant Z is the *normalization factor* also known as the *partition function* to ensure that the distribution p sums to 1. This summation is over the exponentially many possible assignments to \mathbf{x} and \mathbf{y} , and is defined as:

$$Z = \sum_{\mathbf{x}, \mathbf{y}} \prod_{a \subset F} \Psi_a(\mathbf{x}_a, \mathbf{y}_a). \quad (2)$$

We will also assume that each factor (or local function) is represented in an exponential form and is parameterized by θ as:

$$\Psi_a(\mathbf{x}_a, \mathbf{y}_a) = \exp \left\{ \sum_k \theta_{ak} f_{ak}(\mathbf{x}_a, \mathbf{y}_a) \right\}, \quad (3)$$

where θ_a is some real-valued parameter vector, and f_a is some set of feature functions or sufficient statistics.

Given a factorization of a distribution p as in Eq. (1), an equivalent Markov network can be constructed by connecting all pairs of variables that share a local function; p is therefore Markov with respect to the graph defined by this network, because the conditional distribution $p(x_s | \mathbf{x}_{\mathcal{N}(s)})$ is a function only of variables that appear in the Markov blanket. So, if p factorizes according to G , then p is Markov with respect to G and vice versa, as long as p is strictly positive. More formally, this theorem is stated as (Besag, 1974; Hammersley and Clifford, 1971):

Theorem 1 (Hammersley–Clifford). *If p is a strictly positive distribution, and G is an undirected graph that indexes the domain of p , then p is Markov with respect to G if and only if p factorizes according to G .*

In order to state models precisely, the factorization defined in Eq. (1) can be represented directly by means of a factor graph (Kschischang et al., 2001). A factor

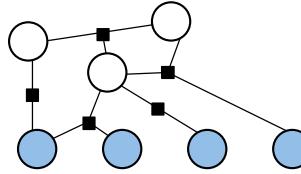


Fig. 2. A general CRF as a factor graph. The circles (shaded and unshaded) are the random variable nodes and the dark shaded boxes are factor nodes.

graph is a bipartite graph $G = (V, F, E)$ where the variable node $v_s \in V$ is connected to a factor node $\Psi_a \in F$ as long as v_s is an argument to Ψ_a . A graphical example of a factor graph for a general CRF is shown in Fig. 2. The circles in Fig. 2 are the random variable nodes and the dark shaded boxes are factor nodes. The factor graph presents the factorization of the model unambiguously.

2.1.1. Undirected graphical models for labeling

We define the labeling (or classification) problem as one of predicting a single discrete class variable y given a vector of features $\mathbf{x} = (x_1, x_2, \dots, x_K)$. A basic well-known undirected graphical model-based classifier is *logistic regression* which is motivated by the assumption that the log probability, $\log p(y|\mathbf{x})$, of each label class is a linear function of \mathbf{x} , plus a normalization constant, leading to the conditional distribution:

$$p(y|\mathbf{x}) = \frac{1}{Z} \exp \left\{ \theta_y + \sum_{j=1}^K \theta_{y,j} x_j \right\}, \quad (4)$$

where $Z(\mathbf{x}) = \sum_y \exp \left\{ \theta_y + \sum_{j=1}^K \theta_{y,j} x_j \right\}$ is a normalizing constant and θ_y is a bias weight. Equation (4) can be further simplified by dropping the restriction to use one weight vector per class ($\theta_{y,j}$), and use a different notation where a single set of weights is shared across all the classes. To do this, we define a set of feature functions that are nonzero only for a single class. These feature functions can be defined as $f_{y',j}(y, \mathbf{x}) = \mathbf{1}_{y'=y} x_j$ for the feature weights and $f_{y'}(y, \mathbf{x}) = \mathbf{1}_{y'=y}$ for the bias weights. The notation $\mathbf{1}_{z=z'}$ is the indicator function of a variable z which takes the value 1 when $z = z'$ and 0 otherwise.

This way, f_k can be used to index each feature function $f_{y',j}$, and θ_k can be used to index its corresponding weight $\theta_{y',j}$. The logistic regression model can therefore be rewritten using this notation as:

$$p(y|\mathbf{x}) = \frac{1}{Z} \exp \left\{ \theta_y + \sum_{k=1}^K \theta_k f_k(y, \mathbf{x}) \right\}. \quad (5)$$

2.1.2. General CRF definition

Although there are specialized CRFs designed with particular choices of feature functions in order to facilitate inference (e.g., the linear-chain CRF uses the forward–backward algorithm), more general CRFs are constructed over any

arbitrary factor graph, using more general (usually approximate) inference algorithms.

Definition 1. If G is a factor graph over Y , then $p(y|x)$ is a conditional random field if for any fixed \mathbf{x} , the distribution $p(y|x)$ factorizes according to G .

Every conditional distribution $p(y|x)$ is therefore a CRF for some factor graph. If $F = \{\Psi_a\}$ is the set of factors in G , and each factor takes the exponential family form three so that the resulting conditional distribution takes the form:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{\Psi_A \in G} \exp \left\{ \sum_{k=1}^{K(A)} \theta_{ak} f_{ak}(\mathbf{y}_A, \mathbf{x}_A) \right\}. \quad (6)$$

The factors of G can be partitioned into $\mathcal{C} = \{C_1, C_2, \dots, C_p\}$, where each C_p is a *clique template* whose parameters are tied. Each clique template C_p is a set of factors with a set of sufficient statistics $\{f_{pk}(\mathbf{y}_p, \mathbf{x}_p)\}$ and parameters $\theta_p \in \mathbb{R}^{K(p)}$. The CRF can be written in terms of its clique partitioning as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{C_p \in \mathcal{C}} \prod_{\Psi_c \in C_p} \Psi_c(\mathbf{x}_c, \mathbf{y}_c; \theta_p), \quad (7)$$

and every factor can be parameterized as:

$$\Psi_c(\mathbf{x}_c, \mathbf{y}_c; \theta_p) = \exp \left\{ \sum_{k=1}^{K(p)} \theta_{pk} f_{pk}(\mathbf{y}_c, \mathbf{x}_c) \right\}, \quad (8)$$

the normalization function Z is of the form:

$$Z(\mathbf{x}) = \sum_y \prod_{C_p \in \mathcal{C}} \prod_{\Psi_c \in C_p} \Psi_c(\mathbf{x}_c, \mathbf{y}_c; \theta_p). \quad (9)$$

An important aspect in defining a general CRF lies in specifying the repeated structure and in parameter sharing.

2.2. Inference

Although exact inference algorithms exist for general graphs, they require exponential time in the worst case, but they can still be useful for graphs that occur in practice. Efficient inference is critical for CRFs both during training and during label prediction on new/test data. The two inference problems that occur are (i) label prediction on new input \mathbf{x} after training, using the most likely $\mathbf{y}^* = \operatorname{argmax}_y p(\mathbf{y}|\mathbf{x})$; and (ii) parameter estimation of $\theta = \{\theta_k\}$ which requires the computation of the marginal distribution $p(y, \mathcal{N}(y) \setminus y | \mathbf{x})$ and the normalizing function $Z(\mathbf{x})$. One issue raised by these closely related problems is that when approximate inference methods are implemented, there can be complex interactions between the inference procedure and the parameter estimation procedure. Their strong dependence on each other strongly influences the choice of inference algorithm.

An example of an exact inference algorithm is the junction tree algorithm which successively clusters variables until the graph becomes a tree. The marginals can therefore be computed on the resulting equivalence tree using exact inference algorithms, specific to trees. But there are certain complex graphs for which the junction tree algorithm fails because the clusters in the equivalence tree can be very large, again resulting in exponential time in the worst case. Two classes of approximate algorithms very popularly used with CRFs are: *Markov Chain Monte Carlo* or *sampling algorithms* and *variational algorithms*. We will briefly examine one algorithm from each class of methods.

2.2.1. Markov Chain Monte Carlo (MCMC)

MCMC methods work by constructing a Markov chain, whose state space is the same as that of Y , so that when the chain is simulated for a long time, the distribution over the states of the chain is approximately $p(y_s|x)$. This way, approximate statistics (expectation, mode, etc.) of different functions can be evaluated from the samples drawn.

In the context of CRFs, such approximate expectations can then be used to approximate the gradient (and maybe other quantities) required for learning. Gibbs sampling is a simple example of an MCMC method where, in each iteration of the Gibbs sampling algorithm, each variable is resampled individually, keeping all of the other variables fixed. Suppose that we currently have a sample \mathbf{y}^j from iteration j , to generate the next sample \mathbf{y}^{j+1} we do the following:

1. Set $\mathbf{y}^{j+1} \leftarrow \mathbf{y}^j$.
2. For each $s \in V$, resample variable s . Sample \mathbf{y}_s^{j+1} from the distribution $p(y_s|\mathbf{y}_{\setminus s}, \mathbf{x})$.
3. Return the new value of \mathbf{y}^{j+1} .

Using the Gibbs sampling decomposition, the CRF defined in Eq. (7) can be rewritten as:

$$p(y_s|\mathbf{y}_{\setminus s}, \mathbf{x}) = \kappa \prod_{C_p \in \mathcal{C}} \prod_{\Psi_c \in C_p} \Psi_c(\mathbf{x}_c, \mathbf{y}_c; \theta_p), \quad (10)$$

where κ is the normalizing constant. It turns out that $p(y_s|\mathbf{y}_{\setminus s}, \mathbf{x})$ is significantly easier to compute than $p(\mathbf{y}|\mathbf{x})$ since the computation of κ requires a summation of overall possible values of y_s rather than all assignments to the complete vector \mathbf{y} . Although Gibbs sampling is easy to implement, it could perform poorly if there are strong dependencies in $p(\mathbf{y}|\mathbf{x})$ which is often the case in real-life data such as in image labeling. MCMC methods are not typically run for CRF problems because the parameter estimation by maximum likelihood requires calculating marginals many times. One MCMC chain would be run for each training example, for each parameter setting that is visited in the course of a gradient descent algorithm. Since MCMC chains can take thousands of iterations to converge, this can be computationally prohibitive.

2.2.2. Variational mean field

Variational algorithms are algorithms that convert the inference problem into an optimization problem, by attempting to find a simple distribution that most closely matches the intractable distribution of interest. They can be much faster than sampling-based procedures, but they tend to be biased, i.e., they tend to have a source of error that is inherent to the approximation, and cannot be easily lessened by additional computation times. Despite this, variational algorithms can be useful for CRFs because parameter estimation requires performing inference many times, so a fast inference procedure is vital to efficient training.

The CRF inference problem occurring during label prediction on new/test data can be phrased as a *maximum a priori* (MAP) problem such that the label prediction on new input \mathbf{x} after training can be obtained by the most likely $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$ while a closely related adaptation of the MAP problem known as the *maximum posterior marginals* (MPM) can be obtained by $y_i^* = \operatorname{argmax}_{\mathbf{y}} p(y_i|\mathbf{x})$.

Given the general factorization of a distribution p , the equivalent energy function of Eq. (6), can be expressed as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(-E(\mathbf{y}|\mathbf{x})), \quad (11)$$

$$E(\mathbf{y}|\mathbf{x}) = \sum_{\phi \in \Phi} \phi(y_\phi|\mathbf{x}). \quad (12)$$

We define several notations, useful in describing variational mean field: (i) an unnormalized distribution over labels as $\tilde{P}(\mathbf{y}) = \exp(-E(\mathbf{y}))$; (ii) a variational distribution over labels $Q(\mathbf{y})$; where any variational distribution $Q(\mathbf{Z})$ is an approximation of a posterior distribution $P(\mathbf{Z}|\mathbf{X})$ and $\mathbf{Z} = \{z_1, z_2, \dots, z_n\}$ is a set of unobserved variables. Typically, $Q(\mathbf{Z})$ is selected to belong to a family of distributions of simpler form than $P(\mathbf{Z}|\mathbf{X})$. The lack of similarity between them is measured in terms of a dissimilarity function and hence inference is performed by selecting the distribution that minimizes this dissimilarity function. Variational mean field inference uses the Kullback Leibler divergence (KL-divergence) of P from Q as the choice of dissimilarity function and this choice makes this minimization tractable.

We start the process of describing our variational probabilistic inference with an over-simplistic objective function:

$$J(Q) = -KL(Q_y \| P_{y|x}) + \log P(x), \quad (13)$$

where Q is the variational distribution over the hidden labels as described previously and $KL(\cdot)$ is the KL-divergence:

1. If $Q(y) = P(y|x)$, the posterior probability over the hidden label variables y , then we recover the desired log-marginal $\log P(x)$.
2. $\log P(x) \geq J(Q)$ for all Q , so that the objective function is always a lower bound on the desired quantity.
3. The slack in the bound is given by the KL-divergence between Q and the true posterior.

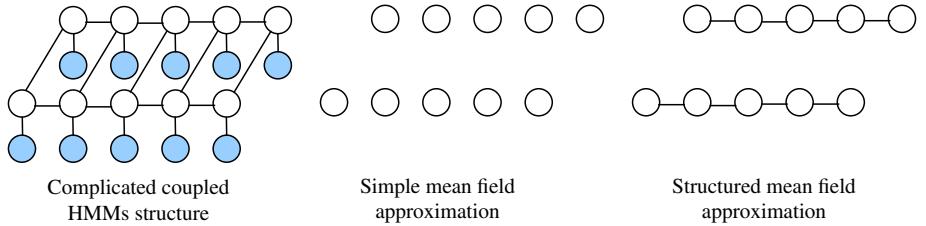


Fig. 3. A complicated graphical structure on the left represented by simpler variational approximations in the center and on the right. The distance between the structure on the left and the one in the middle (or the right) is the KL-divergence $KL(Q_y||P_{y|x})$; P is the posterior distribution of the left structure and Q is the variational distribution of either the center or the right approximation.

This objective function can be rewritten without reference to the posterior or the marginal as:

$$J(Q) = \sum_y Q(y) \log P(x,y) + H(Q), \quad (14)$$

where $H(Q)$ is the (Shannon) entropy of Q . Taking the logarithm of the joint distribution rather than the conditional distribution is attractive because of the factorization (same as in Eqs. 7–9):

$$\log p(\mathbf{x}, \mathbf{y}) = \log \prod_{c \in C} \Psi_c(\mathbf{x}_c, \mathbf{y}_c) = \sum_{c \in C} \log \Psi_c(\mathbf{x}_c, \mathbf{y}_c). \quad (15)$$

Although the variational formulation in Eq. (14) is exact it is not (yet) manageable, hence we can restrict the maximization of Eq. (13) to some manageable class of distributions Q , and the simplest choice of a completely factored or *mean field* is the choice

$$Q(y) = \prod_i Q_i(y_i). \quad (16)$$

Figure 3 shows how a complicated graph structure such as a coupled hidden Markov model (HMM) graph is approximated with a simple factored model.

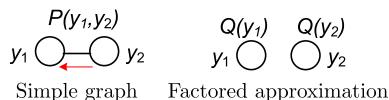
So how do we optimize the simple variational distribution? We derive the mean field update equations by:

$$\text{Maximizing } J(Q) = \sum_y Q(y) \log P(x,y) + H(Q) \text{ same as Eq. (14)}$$

with respect to each of the marginals $Q_i(y_i)$ in

$$Q(y) = \prod_i Q_i(y_i) \text{ same as Eq. (16)}$$

while keeping the remaining marginals fixed. (17)



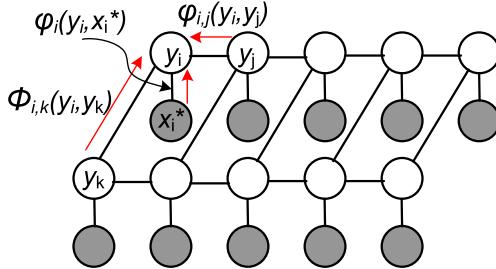


Fig. 4. Variational mean field updates on the coupled HMM graphical structure from Fig. 3.

In the example shown in Fig. 2.2.2, the updates are made “locally” by geometrically averaging the ignored dependencies with respect to the other component distributions so that:

$$Q_1(y_1) \leftarrow \frac{1}{Z_1} \times \exp \left(\sum_{y_2} Q_2(y_2) \log P(y_1, y_2) \right), \quad (18)$$

where Z_1 normalizes the right-hand side across x_1 . Hence, each y_i only sees the “mean effect” of its dependencies. Extrapolating this example to the coupled HMM graph in Fig. 4 gives an update equation involving (i) ψ_i —the evidence at i ; (ii) ψ_{ij} —the within chain dependencies; and (iii) ϕ_{ik} —the between chain dependencies:

$$\begin{aligned} Q_i(y_i) &\leftarrow \frac{1}{Z_i} \times \psi_i(y_i, x_i^*) \times \exp \left(\sum_{y_j} Q_j(y_j) \log \psi_{ij}(y_i, y_j) \right) \\ &\quad \times \exp \left(\sum_{y_k} Q_k(y_k) \log \phi_{ik}(y_i, y_k) \right). \end{aligned}$$

The variation mean field inference will yield good accuracy when there is weak dependence (or near independence) among the nodes in the original graph, so that it proves useful even for large systems as long as there are only weak interactions within the system. The methods begin to fail and yield poor approximations when there are strong dependencies. Other more realistic version of this variational approach includes structured mean field and generalized mean field methods.

3. Scene parsing

Semantic segmentation (i.e., the task of assigning each pixel of a photograph to a semantic class label) can be accomplished using the CRF model which is typically the subdivision of an image into small nonoverlapping elements (pixels, patches, or small superpixels). It then learns and evaluates the likelihood of each element as belonging

to one of the semantic classes (unary terms) and combines these likelihoods with pairwise terms that encourage neighboring elements to take the same labels, and in this way it propagates the information from elements that are certain about their labels to uncertain ones. The appeal of this CRF model is the availability of efficient MAP inference methods, which can be exact for two-label problems with submodular pairwise terms and gets very close to global optima for many practical cases of multi-label segmentation. In the rest of this section, we explain the steps involved in labeling a scene image using the basic CRF model and present several variations and extensions to the basic CRF.

3.1. Scene features

Initially, the image of a scene is made up simply of a 2D array of RGB pixels. To facilitate processing, the image can be re-represented in different forms such as superpixels which correspond to small, nearly uniform regions in the image. The use of superpixels was initially found to improve the computational efficiency of a labeling task and allowed slightly more complex statistics to be computed for enhancing the knowledge of the image structure (Nwogu and Corso, 2008; Hoiem et al., 2007; Gould et al., 2009). Another common representation of 2D images for labeling purposes is the use of rectangular grids of patches at a single scale, thus associating a hidden class label with each patch (Verbeek and Triggs, 2007). More recently, for improved accuracy, the computer vision community has gone full circle so that labels are now being assigned directly to image pixels, rather than to other higher-level representations of the image (Krähenbühl and Koltun, 2011). It can be argued that since each region (superpixel or patch) takes only one semantic label, the regions have to be small enough to not straddle class boundaries too often. Thus, the amount of visual information inside the superpixel is limited. The best performing CRF models therefore consider wider local context around each superpixel, but since the class boundaries are not known in advance, the support area over which such context information is aggregated is not adapted. For this reason, such context-based descriptors are argued to not necessarily allow reliable classification. Hence, more recent models attempt to perform segmentation and labeling simultaneously starting directly from the image pixels (see Fig. 5).

Any region in a scene image could theoretically be generated by any surface, so in order to determine the most likely surface (from a given class of surfaces), we use all of the available cues such as the material, location, texture gradients, shading, vanishing points, etc. as cues. In Table 1, we list some of the statistics used for classification.

3.1.1. Color

By modeling color, the material and objects that correspond to either a geometric or semantic class can be implicitly inferred, thus making color a very powerful cue for labeling scenes. For instance, the sky is usually a particular shade of blue or is white, while segments that are green are often either grass or other types of foliage. Typically, color is represented in one or more color spaces and some examples include the RGB or HSV color spaces. RGB highlights the “blueness” or “greenness” of an image



Fig. 5. An example of two different scene labeling tasks—geometric and semantic labelings.

Table 1
Cues for scene image labeling

Surface cues

Location and shape

Location: Normalized y , mean

Location: Normalized y , 10th and 90th percentile

Location: Normalized y wrt estimated horizon, 10th, 90th percentile

Location: Whether segment is above, below, or straddles estimated horizon

Shape: Number of superpixels in segment

Shape: Normalized area in image

Color

Mean values of segment color (multiple color spaces)

Histogram of a single channel of the color space, e.g., hue

Texture

Mean absolute response (multiple filters)

Histogram of maximum responses (multiple bins)

Histogram of vector quantized SIFT descriptors

region, while HSV allows perceptual color attributes such as hue and “grayness” to be measured.

3.1.2. Location

Location here refers to the position in the 2D image. Location provides a strong cue for the presence of different surfaces in an image. As might be expected, ground surfaces such as grass, bodies of water, etc. tend to be located in the lower position in the image, while sky tends to be high. The x -position in the image tells little about the surfaces or regions in the image. Pixel y -locations are typically normalized by the height of the image so that statistics such as the mean, the 10th and 90th percentiles y -position of the segments in the image can be computed. If superpixels are used, it is also useful to measure the number of superpixels and pixels, normalized by the total image area, for each segment.

3.1.3. Texture

Texture provides a cue for both the geometric and semantic class of a segment through its relationship to materials and objects in the world. Texture is typically represented via applying a subset of filter banks. Some of these filter banks include the Gabor filter banks, Gaussian derivative filter banks, and more specific filters such as those designed by [Leung and Malik \(2001\)](#). Standard texture measuring filters banks should contain oriented and blob filters for multi-scale, multi-orientation edge extractors, multi-bar extractors, Gaussian filters, and Laplacian of Gaussian filters. The texture representation could include the absolute filter responses of each filter and the histogram (over pixels within a segment) of maximum responses. Another form of measuring texture responses especially in patch-based techniques involves the computation SIFT descriptors ([Krähenbühl and Koltun, 2011](#)) (e.g., the 128-dimensional) of the patch and then vector quantizing the descriptors using nearest-neighbor assignment against a k -word texton dictionary learned by k -means clustering of all patches in the training dataset.

3.2. Datasets

Datasets are an integral part of contemporary scene labeling research. They have been the chief reason for the considerable progress in the field, not just as sources of large amounts of training data, but also as means of measuring and comparing performance of competing algorithms. The goal of this paper subsection is to take stock of the current state of recognition datasets used for different forms of scene labeling tasks. Some examples of scene labeling datasets include Corel Stock Photos and 15 Scenes ([Oliva and Torralba, 2001](#)) which are photographs from professional collections, embracing visual complexity found in the real world, Caltech-101 ([Fei-Fei et al., 2004](#)) which contains 101 objects mined using Google and cleaned by hand, MSRC ([Winn et al., 2005](#)) and LabelMe ([Russell et al., 2008](#)) collected by researchers in an attempt to counter the Caltech-like single-object-in-the-center mentality. These two datasets are at the heart of scene parsing algorithms since they contain many images of complex outdoor scenes with many objects. Other datasets such as the Stanford background dataset ([Gould et al., 2009](#)) are an aggregation of

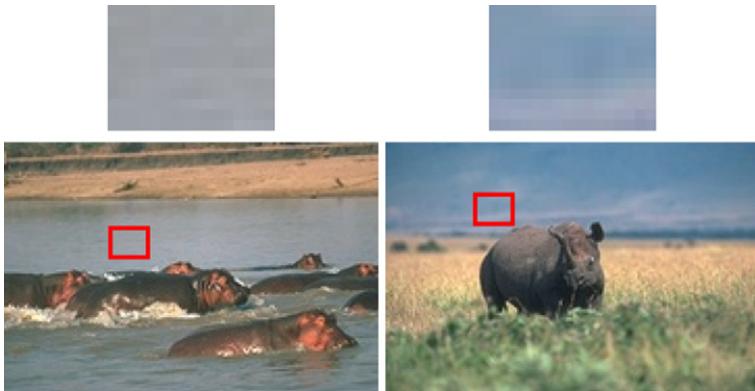


Fig. 6. *Top:* Two small image patches that are difficult to label based on local information alone. *Bottom:* Images containing the patches. The global context clarifies what the patches are (left: water; right: sky). Image obtained from [He et al. \(2004\)](#).

the previous datasets such as MSRC and Corel along with a number of additional images. [Torralba and Efros \(2011\)](#) reviewed these datasets and more, and noted the existence of some very biased sets with virtually no generalizations, e.g., the Caltech-101 and MSRC datasets.

3.3. An end-to-end example of CRF scene labeling implementation

A patch-based CRF model incorporates four-neighbor couplings between patch labels. The local image content of each patch can be encoded using texture, color, and position or location descriptors as described in Section 3. For patch-based representations, a useful method for encoding position is to overlay the image with an $m \times m$ grid of cells (e.g., $m = 8$) and the index of the cell in which the patch falls becomes its position feature. The CRF observation functions are simple linear functions of the three vectors representing texture, color, and position. In general, the three modalities are modeled as being independent given the patch label. A naive Bayes model of the image omits the four-neighbor couplings and thus assumes that each patch label depends only on its three observation functions so that parameter estimation reduces trivially to counting observed visual word frequencies for each label class and feature type. On the MSRC 9-class image dataset, this model returned an average classification rate of 67.1%, indicating that such isolated appearance models do not suffice for reliable patch labeling.

Many CRF models for labeling take the global image context into account by including observation functions based on image-wide histograms of the visual words of their patches. The hope is that this will help to overcome the ambiguities that arise when patches are classified in isolation (Fig. 6). To this end, a conditional model is defined for patch labels that incorporate both local patch level features and global aggregate features.

Let $x_i \in \{1, \dots, C\}$ denote the label of patch i , \mathbf{y}_i denote the W -dimensional concatenated binary indicator vector of its three visual words, and \mathbf{h} denote the

normalized histogram of all visual words in the image, i.e., $\sum_{\text{patches}_i} y_i$ normalized to sum to one. The conditional probability of the label x_i can therefore be modeled as:

$$p(x_i = l | \mathbf{y}_i, \mathbf{h}) \propto \exp \left(- \sum_{w=1}^W (\alpha_{wl} y_{iw} + \beta_{wl} h_w) \right), \quad (19)$$

where α_{wl} and β_{wl} are $W \times C$ matrices of coefficients to be learned. This can be viewed as a multiplicative combination of a local classifier based on the patch-level observation y_i and a global context or bias based on the image-wide histogram \mathbf{h} .

The correlations among spatially neighboring patch labels can be accounted by adding couplings between the labels of neighboring patches to the single patch model in the equation above. Let X denote the collection of all patch labels in the image and Y denote the collected patch features. Then the CRF model for the coupled patch labels is:

$$p(X|Y) \propto \exp(-E(X|Y)), \quad (20)$$

$$E(X|Y) = \sum_i \sum_{w=1}^W (\alpha_{wl} y_{iw} + \beta_{wl} h_w) + \sum_{i \sim j} \phi_{ij}(x_i, x_j), \quad (21)$$

where $i \sim j$ is the set of all adjacent four-neighbor pairs of patches i, j . A form of the pairwise potential can be given as $\phi_{ij}(x_i, x_j) = \gamma_{x_i, x_j}[x_i \neq x_j]$. The term $[\cdot]$ is the indicator function which resolves to one if its argument is true and zero otherwise; γ_{x_i, x_j} is a general symmetric weight matrix that needs to be learned. Another form of a pairwise potential suitable for this model is $\phi_{ij}(x_i, x_j) = (\sigma + \tau d_{ij})[x_i \neq x_j]$. This second potential is designed to favor label transitions at image locations with high contrast. The term d_{ij} is a type of similarity measure over the appearance of the patches and is given by $d_{ij} = \exp(-\|z_i - z_j\|^2/(2\lambda))$ where $z_i \in \mathbb{R}^3$, denoting the average RGB value in the patch. Also, $\lambda = \langle \|z_i - z_j\|^2 \rangle$ is the average L_2 norm between neighboring RGB values in the image.

To estimate the conditional random field, the given models $p(X|Y)$ above are usually trained by maximizing the log-likelihood of correct classification of the training data, $\sum_n^N \log p(X_n|Y_n)$ which requires labeled data, a collection of N pairs $(X_n, Y_n)_{n=1, \dots, N}$. We maximize the log-likelihood of the model:

$$L = \log \sum_X p(X|Y) = \log \left(\sum_X \exp(-E(X|Y)) \right). \quad (22)$$

To find the maximum likelihood parameter estimates using gradient descent we need to calculate partial derivatives with respect to each parameter θ so that:

$$\frac{\partial L}{\partial \theta} = - \sum_X \frac{\partial E(X|Y)}{\partial \theta}. \quad (23)$$

The necessary marginals are approximated using Loopy Belief Propagation (LBP). Verbeek and Triggs (2007) extended this patch-based model to label new images from partially labeled images, where the training data consisted of many

unlabeled nodes. They extended the algorithm to marginalize the unknown labels so that the log-likelihood of the known ones could be maximized by gradient ascent. Loopy Belief Propagation was also used to approximate the marginals needed for the gradient and log-likelihood calculations. The Bethe free-energy approximation to the log-likelihood was closely monitored to control the step size. They successfully showed that effective models could be learned from fragmentary labelings and that incorporating top-down aggregate features significantly improved the patch-based image segmentations and their labeling. In practice, the energy model using the term with the similarity measure d_{ij} performed better than its learned symmetric weight matrix γ_{x_i, x_j} counterpart.

4. More recent implementations of CRF scene labelings

4.1. Hierarchical CRF for scene labeling

Kumar and Hebert (2005) as well as Yang and Förstner (2011) proposed a hierarchical CRF model for labeling scenes. In this hierarchical approach, each layer is modeled as a CRF in order to model interactions in images at two different levels. The proposed two-layer hierarchical field model for scene labeling is shown in Fig. 7.

The first layer models short-range interactions among the nodes such as label smoothing for pixelwise labeling, or geometric consistency among parts of an object. The second layer models the long-range interactions between groups of nodes corresponding to different coherent regions or objects. Thus, this layer can take into account interactions between scene regions such as sky-and-water, trees-and-road, etc.

For layer 1, the conditional distribution of the labels given the observed data— $P(x^{(1)}|y)$ —is directly modeled as a homogeneous pairwise conditional random field as:

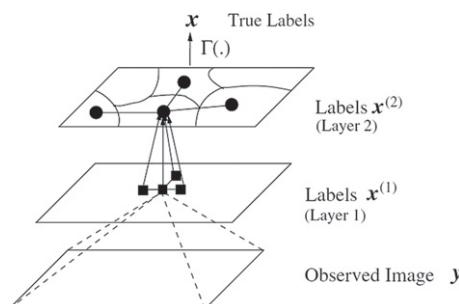


Fig. 7. An illustration of the two-layer hierarchical CRF for contextual classification. Squares and circles represent nodes at the two layers. For clarity, only one node along with its neighbors is shown. Layer 1 models short-range interactions while layer 2 long-range dependencies in images. The true labels x are obtained from the top layer by a simple replication mapping $\Gamma(\cdot)$. The partition shown in the top layer is not necessarily a partition on the image. Image obtained from Kumar and Hebert (2005).

$$P(x^{(1)}|y) = \frac{1}{Z} \prod_{i \in S^{(1)}} i \in S^{(1)} \phi\left(x_i^{(1)}, y\right) \prod_{i, j \in \mathcal{N}_i} \psi\left(x_i^{(1)}, x_j^{(1)}, y\right), \quad (24)$$

where Z is a normalizing constant or the partition function, $S^{(1)}$ is the set of nodes in layer 1, \mathcal{N}_i is the set of neighbors of node i , $\phi(x_i^{(1)}, y)$ and $\psi(x_i^{(1)}, x_j^{(1)}, y)$ are the unary and the pairwise potentials.

The formulation of the conditional field for layer 2 can be obtained in the same way as in layer 1 but changing the values of the observed variables and the new set of nodes to $S^{(2)}$. Starting with parameter learning in layer 1, since the labels at this layer are not known, pseudo-labels $x^{(1)}$ are assigned on S using the true labels x . For image labeling, since the nodes at both layers take their labels from the same set, one can assume the pseudo labels to be the same as the true labels. The top layer will eventually refine the label estimates from the layer below and the directed connections incorporate the transition probabilities from the noisy labels to the true labels. Gradient ascent is used to learn the parameters of the conditional field in layer 1. Expectations are estimated using the pseudo-marginals returned by Loopy Belief Propagation (LBP) algorithm. The transition probability matrices are assumed to be the same for all the directed links in the graph to avoid overfitting. For inference over the entire graph, again LBP was used to find the maximum marginal estimates of the labels on the image nodes.

4.2. Fully connected CRF for scene labeling

Zhang and Chen (2012) proposed a unified model to augment pixelwise CRFs in order to capture object spatial relationships. To this end, they used a fully connected CRF, having an edge for each pair of pixels on the image grid. The edge potentials were defined to capture the spatial information and preserve the object boundaries at the same time. Because traditional inference methods such as belief propagation and graph cuts are impractical in such a case where billions of edges are defined, they proposed an efficient inference algorithm that converged in a few seconds on a standard resolution image.

Using the standard CRF definitions, the labeling X of an image can be obtained with a maximum a posteriori (MAP) estimation of the following conditional log-likelihood:

$$\log P(\mathbf{X}|\mathbf{I}) = \sum_{i \in V} \psi_i(x_i) + \sum_{i \in V, j \in \mathbb{N}_i} \psi_{ij}(x_i, x_j) - Z(\mathbf{I}). \quad (25)$$

The scene labeling problem is thus formulated as a fully connected CRF defined over the image pixels. The conditional log-likelihood is the same as above except that the edges now are defined over all pairs of pixels in the image. Hence the neighborhood of any pixel i is defined with all other pixels $\mathbb{N}_i = V$. The unary potential $\psi_i(x_i)$ is the same as the standard definition usually computed with the texture, color, and location features extracted from a local region around i . But the edge or pairwise potential $\psi_{ij}(x_i, x_j)$ here is a combination of the color contrast $\varphi_{ij}(x_i, x_j)$ and the spatial relationships between two categories, $\phi_{ij}(x_i, x_j)$:

$$\psi_{ij}(x_i, x_j) = \phi_{ij}(x_i, x_j)\varphi_{ij}(x_i, x_j). \quad (26)$$

To obtain a MAP estimation of their CRF model, they adopted the quadratic programming (QP) relaxation method which has been shown to perform comparable to other inference methods, such as LP relaxation and tree-reweighted message passing. The labeling problem that maximizes the conditional probability can be viewed as an integer program by adding a binary variable $\mu_i(x_i)$, which indicates whether a pixel i is labeled with x_i :

$$\mu_i(x_i) = \begin{cases} 1 & \text{if } X_i = x_i, \\ 0 & \text{otherwise.} \end{cases} \quad (27)$$

The MAP estimation for the conditional probability in Eq. (25) can be formulated as a quadratic integer program, which is further relaxed to the following quadratic program:

$$\begin{aligned} \max_{\mu} \quad & \sum_{i; x_i} \psi_i(x_i) \mu_i(x_i) + \sum_{i, j; x_i, x_j} \psi_{ij}(x_i, x_j) \mu_i(x_i) \mu_j(x_j) \\ \text{s.t.} \quad & \sum_i \mu_i(x_i) = 1, \\ & 0 \leq \mu_i(x_i) \leq 1. \end{aligned} \quad (28)$$

This QP relaxation has been proven to be a tight relaxation, and solves exactly the original MAP problem. They initialized μ with the unary probabilities, and iteratively updated it with the gradient of the objective function Q . But computing the gradient of a fully connected CRF is a major bottleneck to an approach such as this as a naive implementation would have a computational complexity of $O(N^2)$ for computing the gradient term for all pixels, where N is the number of pixels in the image. The computation is simplified by propagating messages to similar color pixels for the same category while for different categories, messages are propagated to pixels of different colors, which are more likely to originate from different regions/objects in the image. Image filtering is used to solve the message propagation problem and filtering can be greatly accelerated with Fast Fourier Transform (FFT), which reduces the complexity to $O(N \log N)$. Image filtering on both space and color dimensions is also called Bilateral Filtering and the spatial filter here is a Gaussian. For learning the edge potential, the co-occurrence distribution of two categories is obtained by counting the number of times any two categories appear in the same image and, their relative spatial distribution is obtained by observing the image

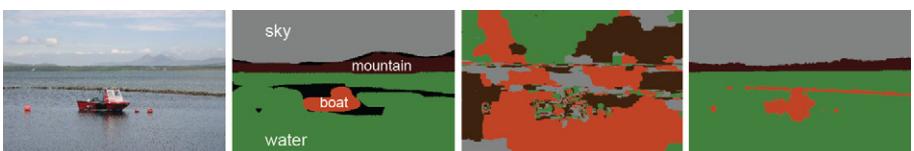


Fig. 8. An example comparing the labeling results between a patch-based CRF model and a fully connected CRF model. From left to right are the original image, ground-truth labels, patch-based CRF labeling results, and fully connected pixel-based CRF labeling results. Image obtained from [Zhang and Chen \(2012\)](#).

which contains the two categories being investigated and counting the frequency at which they occur at relative positions. This was accomplished efficiently using cross-correlation over the pixelwise ground truth.

This fully connected CRF thus encoded the spatial relationships among different objects and preserved object contours at the same time and currently provides one of the highest accuracy labeling results on several benchmark scene labeling/object detection datasets (see Fig. 8).

5. Conclusion and future directions

Conditional random fields for scene labeling offer a unique combination of properties: discriminatively trained models for segmentation and labeling; combination of arbitrary, overlapping, and agglomerative observation features from neighboring nodes in the graph; efficient training and decoding based on optimization techniques such as dynamic programming; and parameter estimation guaranteed to find the global optimum.

In the future, it will be useful to continue to explore the use of variational approximations to relax some of the assumptions made in many scene labeling CRF models. Also, with the advent of advanced learning methods for evaluating deep networks (Salakhutdinov and Hinton, 2012) it will be useful to utilize full MCMC sampling methods in the parameter learning and inference steps, rather than using data-driven heuristics as typically done. Also, it will be useful to construct higher levels of CRF hierarchies and develop efficient ways of learning the parameters of the multiple layers simultaneously, to explore the possibility of encoding more complex relations between different scenes in a video, leading to event or activity recognition.

Additional information relating to the use of conditional random fields for labeling scenes can be found in the following articles: Everingham et al. (2010), Felzenszwalb and Huttenlocher (2004), Konishi and Yuille (2000), Kumar and Koller (2010), Levenshtein et al. (2009), Li et al. (2009), Liu et al. (2010), Saxena et al. (2005, 2009), Shotton et al. (2009), Socher et al. (2011), Tighe and Lazebnik (2010), Tu et al. (2005), and Yedidia et al. (2005).

References

- Besag, J., 1974. Spatial interaction and the statistical analysis of lattice systems. *J. Roy. Stat. Soc. Ser. B* 36 (2), 192–236.
- Clifford, P., 1990. Markov random fields in statistics. *Disorder in Physical Systems: A Volume in Honour of John M. Hammersley*. Oxford University Press, pp. 19–32.
- Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2010. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. <<http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html>>.
- Fei-Fei, L., Fergus, R., Perona, P., 2004. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: *CVPRW'04*.
- Felzenszwalb, P.F., Huttenlocher, D.P., 2004. Efficient graph-based image segmentation. *Int. J. Comput. Vis.* 59, 167–181.

- Gould, S., Fulton, R., Koller, D., 2009. Decomposing a scene into geometric and semantically consistent regions. In: ICCV.
- Hammersley, J.M., Clifford, P., 1971. Markov fields on finite graphs and lattices.
- He, X., Zemel, R.S., Carreira-Perpiñán, M.A., 2004. Multiscale conditional random fields for image labeling. In: CVPR, pp. 695–702.
- Hoiem, D., Efros, A.A., Hebert, M., 2007. Recovering surface layout from an image. Int. J. Comput. Vis. 75 (1), 151–172.
- Koller, D., Friedman, N., 2009. Probabilistic Graphical Models: Principles and Techniques. MIT Press.
- Konishi, S., Yuille, A.L., 2000. Statistical cues for domain specific image segmentation with performance analysis. In: CVPR, pp. 1125–1132.
- Krähenbühl, P., Koltun, V., 2011. Efficient inference in fully connected CRFs with Gaussian edge potentials. In: Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems (NIPS). MIT Press, pp. 109–117.
- Kschischang, F.R., Frey, B.J., Loeliger, H.-A., 2001. Factor graphs and the sum-product algorithm. IEEE Trans. Inform. Theory 47 (2), 498–519.
- Kumar, S., Hebert, M., 2005. A hierarchical field framework for unified context-based classification. In: ICCV, pp. 1284–1291.
- Kumar, M.P., Koller, D., 2010. Efficiently selecting regions for scene understanding. In: CVPR, 3217–3224.
- Leung, T., Malik, J., 2001. Representing and recognizing the visual appearance of materials using three-dimensional textons. Int. J. Comput. Vis. 43, 2944.
- Levinstein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K., 2009. Turbopixels: fast superpixels using geometric flows. IEEE Trans. Pattern Anal. Mach. Learn. 31 (12), 2290–2297.
- Li, L.-J., Socher, R., Li, F.-F., 2009. Towards total scene understanding: classification, annotation and segmentation in an automatic framework. In: CVPR, pp. 2036–2043.
- Liu, B., Gould, S., Koller, D., 2010. Single image depth estimation from predicted semantic labels. In: CVPR.
- Nwogu, I., Corso, J.J., 2008. (BP)²: beyond pairwise belief propagation labeling by approximating Kikuchi free energies. In: CVPR.
- Oliva, A., Torralba, A., 2001. Modeling the shape of the scene: a holistic representation of the spatial envelope. Int. J. Comput. Vis. 42 (3), 145–175.
- Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T., 2008. LabelMe: a database and web-based tool for image annotation. Int. J. Comput. Vis. 77 (1–3), 157–173.
- Salakhutdinov, R., Hinton, G.E., 2012. An efficient learning procedure for deep Boltzmann machines. Neural Comput. 24 (8), 1967–2006.
- Saxena, A., Chung, S.H., Ng, A.Y., 2005. Learning depth from single monocular images. In: Advances in Neural Information Processing Systems (NIPS).
- Saxena, A., Sun, M., Ng, A.Y., 2009. Make3D: learning 3D scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI).
- Shotton, J., Winn, J.M., Rother, C., Criminisi, A., 2009. Textonboost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context. Int. J. Comput. Vis. 81 (1), 2–23.
- Socher, R., Lin, C.C.-Y., Ng, A.Y., Manning, C.D., 2011. Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 26th International Conference on Machine Learning (ICML).
- Sutton, C., McCallum, A., 2010. An introduction to conditional random fields, Technical report, University of Edinburgh, UK.
- Tighe, J., Lazebnik, S., 2010. SuperParsing: scalable nonparametric image parsing with superpixels. In: ECCV, vol. 5, 352–365.
- Torralba, A., Efros, A.A., 2011. Unbiased look at dataset bias. In: CVPR’11.
- Tu, Z., Chen, X., Yuille, A.L., Zhu, S.-C., 2005. Image parsing: unifying segmentation, detection, and recognition. Int. J. Comput. Vis. 63 (2), 113–140.
- Verbeek, J., Triggs, B., 2007. Scene segmentation with conditional random fields learned from partially labeled images. In: Advances in Neural Information Processing Systems (NIPS). MIT Press.
- Winn, J., Criminisi, A., Minka, T., 2005. Object categorization by learned universal visual dictionary. In: ICCV.

- Yang, M.Y., Förstner, W., 2011. A hierarchical conditional random field model for labeling and classifying images of man-made scenes. In: ICCV Workshops, pp. 196–203.
- Yedidia, J.S., Freeman, W.T., Weiss, Y., 2005. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inform. Theory* 51 (7), 2282–2312.
- Zhang, Y., Chen, T., 2012. Efficient inference for fully-connected CRFs with stationarity. In: CVPR, pp. 582–589.

This page is intentionally left blank

Shape-Based Image Classification and Retrieval

N. Mohanty^{1,}, A. Lee-St. John^{2,*}, R. Manmatha³,
and T.M. Rath^{1,*}*

¹*FEM-Inc, San Francisco, CA, USA*

²*Department of Computer Science, Mt. Holyoke College, S. Hadley,
MA 01075, USA*

³*Department of Computer Science, University of Massachusetts, Amherst,
MA 01003, USA*

Abstract

Shape descriptors have been used frequently to characterize an image for classification and retrieval tasks. For example, shape features are applied in medical imaging to classify whether an image shows a tumor or not. The patent office uses the similarity of shape to ensure that there are no infringements of copyrighted trademarks. This paper focuses on using machine learning and information retrieval techniques to classify an image as belonging to one of many classes based on its shape. In particular, we compare support vector machines, Naïve Bayes, maximum entropy, linear discriminant analysis, and relevance-based language models for classification. Our results indicate that, on the standard MPEG-7 database, the relevance model outperforms the machine learning techniques and is competitive with prior work on shape-based retrieval with a classification rate of 79.8%. We also show that the relevance model can be easily extended to classify images with multiple labels. The relevance model approach may be used to perform shape retrieval using keywords. Experiments on the MPEG-7 database and a binary version of the COIL-100 database show that good retrieval performance may be obtained using a small number of examples even when there is a large amount of viewpoint change.

Keywords: shape retrieval, shape classification, naïve Bayes, LDA, SVM, relevance models

*This work was done while these authors were at the University of Massachusetts, Amherst.

1. Introduction

The rapid increase in the number of collections of digital images has brought about the need for effective means for searching and classifying these collections. Content-based image retrieval and classification characterizes the image with a set of features and uses these features for classification and retrieval purposes. In some systems these features are used in a manner that allows the user to use an image as a query and find similar images in the database (Latecki et al., 2000; Belongie et al., 2002; Zhang and Lu, 2003). As this interface is frequently inconvenient, researchers (Jeon et al., 2003; Rath and Manmatha, 2003; Barnard et al., 2003; Blei and Jordan, 2003) have come up with an alternate approach where labels associated with a small set of images (training set) are used along with image features to learn the association between words and features. This learned model is then used to automatically classify a vast number of images or retrieve images from a huge collection using a text query without requiring labels to be applied to each image in the collection.

For the purpose of Content-Based Image Retrieval an image is characterized by a set of features that best describe it. These features range from describing color, shape, intensity, edge descriptions, etc. and suitable combinations of simpler features. This paper is based on the idea that the content of many images can be described by the shape of the object in the image. Similar to (Jeon et al., 2003; Rath and Manmatha, 2003; Barnard et al., 2003; Blei and Jordan, 2003) our approach for CBIR involves learning a model for the content of an image (shape in particular) and the annotations associated with it and then using the model to predict the annotations for a new image based on shape. We experiment with a number of machine learning techniques using shape features in order to classify images, including Naïve Bayes, linear discriminant analysis, maximum entropy, support vector machines, and relevance-based language models. Initial work in this direction is presented in Mohanty et al. (2005). Similar to the findings of Mohanty et al. (2005) the relevance model outperforms the remaining techniques. In this paper we also present how the relevance model can be used to classify images with multiple labels.

This paper focuses on classifying, annotating, and retrieving images based on shape. Appropriate shape descriptors are extremely useful features to describe images especially images containing a single object. Classification based solely on shape has been used in a number of different domains. For the purpose of medical imaging, shape helps distinguish between the X-ray images of organs with tumors or unnatural growth from normal ones. Shape-based classification is also useful in classifying visual samples in specific domains such as botanical varieties of plants. Shape features have also been used to distinguish between different objects such as roads, fields, buildings in a topographical shape map. Shape has been used to distinguish between characters in optical character recognition (Belongie et al., 2002), word recognition (Rath and Manmatha, 2003). Shape-based retrieval has been used by the patent office for trademark retrieval (Belongie et al., 2002; Jain and Vailaya, 1998). There are many other applications where the shape of an object has been a key feature in a classification or retrieval task.

There has been a considerable amount of work in finding features that accurately characterize the shape of an object. Most of the work has focused on finding shape

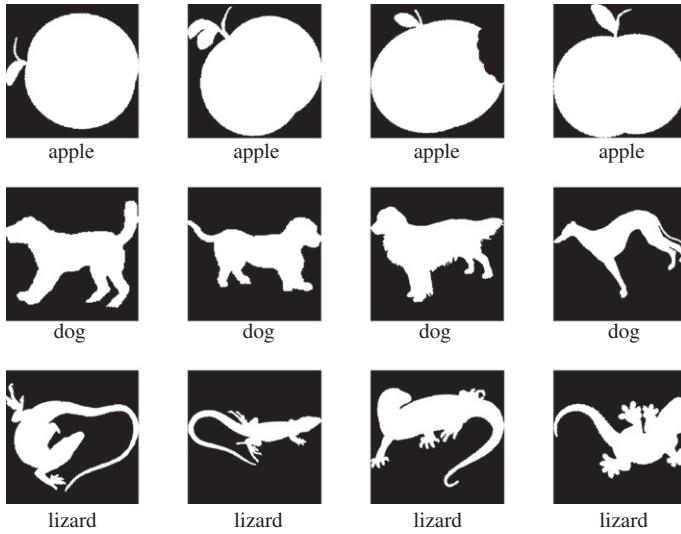


Fig. 1. Sample images and associated labels from the MPEG-7 database.

representations and using them along with image matching techniques in order to retrieve images similar to the query image. We instead focus on using simple shape features, namely Fourier coefficients that describe the contour of the image to learn models describing different class labels which help us retrieve images without requiring image matching.

In the following sections we discuss prior work in this field, then we describe the different statistical models in Section 3. We then go onto describe the different shape descriptors that we experimented along with the required preprocessing in Section 4. In Sections 5 and 6 we discuss the classification and retrieval experiments that we performed. We conclude the paper with Section 8.

2. Prior work

There are a number of shape-based retrieval systems in existence. Jain and Vailaya (1998) described a system that searches a database in two stages. The first stage involves using simple shape features over the entire database in order to obtain a number of plausible retrievals when a query is presented, in a short period of time. The second stage involves removing erroneous matches obtained during the first stage by deformable template matching. Thus their system works by fast pruning followed by complex matching. Latecki et al. (2000) compared different shape descriptors to determine which were stable to rotation and scaling, thus making them useful for classification and retrieval; their approach examined how well the features performed on the MPEG-7 database. Retrieval was evaluated by using each image as a query and finding the number of images retrieved in the top 40 that belonged to the same class as the query image. The best results of 76.45% were obtained

by Latecki et al. (2000) using features that examined the correspondence of visual parts; Mokhtarian et al. (1997) achieved 75.44% classification rate with features that looked at curvature scale space. Belongie et al. (2002) used shape context to examine the similarity of images based on shape and obtained a retrieval rate of 76.51% with the MPEG-7 database. Veltkamp and Hagedoorn (1999) have created a system with highly specialized features to compare shapes.

However, all these attempts have been focused in the direction of shape matching. Super (2004) has taken this a step further and created a system that learns a set of chance probability functions (CPFs) that estimate the probabilities of obtaining a query shape at a certain distance from each training example simply by chance. He attempts nearest neighbor classification of query shape by using the learned CPFs to estimate the chance probabilities of observed distances between query shape and shapes in the database. He obtains an accuracy of 82.69% on the bulls-eye test for the MPEG-7 dataset. It is important to note however that in the process for nearest neighbor classification instances from the training set were also included in the evaluation of the test set. The approach by Adamek (2003) uses an alternative of Multi-Scale Convexity Concavity representation to describe the shape based on performing a 1D discrete cosine transform for each multi-scale contour point feature. They evaluate their performance by using an alternative of the bulls-eye test on the MPEG-7 dataset to obtain a retrieval rate of 86.61%.

Our approach extends prior work by creating a model for each shape and thus actually learning the models for different shapes and using this information for classification and retrieval. Our system uses very simple features along with suitable models which allow it to work in a fast efficient manner and still obtain comparable accuracy rate for classification and retrieval. It is possible to improve on our results by using more specialized features.

3. Classification and retrieval models

Classification of images based on shape involves assigning a label to a given image based on information obtained from other images and their corresponding labels. The goal of each classification technique is to learn a model of each class based on shape features and then use this model to predict the labels for new images. Retrieval involves finding images relevant to a query based on the likelihood that the query would be generated from the model of the image. In this section we describe the various classification and retrieval models that we implemented and provide some intuition of why these models work. In all these models each image is described by a feature vector \vec{s} . The details about how this vector is generated are described in Section 4.

3.1. Naïve Bayes

We implemented the Naïve Bayes (Duda et al., 2001) classifier as a baseline for the different classification and retrieval techniques. The dataset was divided into training and test sets. In the training set, given a particular class c_i we assumed that all the images belonging to that class were described by a normal distribution with a

diagonal covariance matrix. The covariance matrix was the same for different classes. We further assumed uniform priors, i.e., all class labels are represented equally in the training set. Therefore, we can write the discriminant function g_i for each image class c_i as follows:

$$g_i(\vec{s}) = \sum_{k=1}^{|\vec{s}|} \ln p(s_k | c_i), \quad (1)$$

where $|\vec{s}|$ represents the length of the feature vector and $p(s_k | c_i)$ is calculated using the Gaussian distribution for each component s_k of the feature vector \vec{s} . Each instance in the test set is assigned to the class with the $\max\{g_i(\vec{s})\}$ for $1 \leq i \leq \text{NumberOfClasses}$.

3.2. Linear discriminant analysis

Linear Discriminant Analysis (LDA) or Fischer Discriminants (Duda et al., 2001) is a common technique used for dimensionality reduction and classification. LDA provides class separability by drawing a decision region between the different classes. LDA tries to maximize the ratio of the between-class variance and the within-class variance. Given the set of training images represented by their feature vectors, for each class c_i we compute the centroid μ_i and covariance matrix Σ_i . We assume uniform prior class probability as in the case of Naïve Bayes. We thus obtain the within class scatter matrix S_w where we are trying to classify each instance into one of 20 classes

$$S_w = \sum_{i=1}^{20} \Sigma_i. \quad (2)$$

In order to compute the between-class scatter matrix we first compute the center of the entire dataset μ . Let B be a matrix keeping distances between class centers and the center of the entire dataset, $B = [\mu_1 - \mu, \mu_2 - \mu, \dots, \mu_{20} - \mu]$. Then the between-class matrix is

$$S_b = \frac{1}{20} BB^T. \quad (3)$$

Our goal is to maximize the ratio between the between-class variance and the within-class variance

$$\max_w J(w) = \frac{w^T S_b w}{w^T S_w w}. \quad (4)$$

Unlike discriminative machine learning techniques that tend to be useful primarily for binary classification LDA is appropriate for our multi-class domain. The insight behind LDA is that the $k - 1$ dimensional subspace connecting the centroids of each class provides a simpler reduced manifold to make decisions where k is the number of different image classes.

3.3. Support vector machines

3.3.1. SVMs for binary classification

Support vector machines (SVMs) were introduced by Boser et al. (1992) as an approach to binary classification. The input is a collection of training instances and corresponding labels; more precisely, this collection is a set of pairs (s_i, y_i) , where s_i represents the training instance and $y_i \in \{-1, 1\}$ represents the negative or positive label. Using this data, an SVM is constructed by finding *support vectors* that attempt to accurately describe a hyperplane separating the positive instances of the class from the negative.

Assuming a linearly separable space, consider a margin around the hyperplane that measures the distance to each training instance. In addition, allow a positive slack ξ_i for each instance s_i ; this slack accounts for positive instances lying on the negative side of the hyperplane and vice versa. A parameter λ tunes the amount of slack tolerated. SVMs seek to minimize the total margin, while maximizing the individual margins to avoid overfitting by solving the optimization problem

$$\min_{w, b, \xi} \frac{1}{2} w^T w + \lambda \sum_{i=1}^N \xi_i \quad (5)$$

subject to the constraint

$$\forall i, y_i(w^T s_i + b) \geq 1 - \xi_i.$$

However, the features may not admit a terrain that is linearly separable. To accommodate this, a *kernel* function K is used to project to a higher-dimensional space that is separable by a hyperplane. The key property of kernels is the existence of a mapping ϕ from the input space to the higher-dimensional space such that

$$K(x, y) = \phi(x)^T \phi(y). \quad (6)$$

The optimization problem that must be solved is then refined to be

$$\min_{w, b, \xi} \frac{1}{2} w^T w + \lambda \sum_{i=1}^N \xi_i \quad (7)$$

subject to the constraint

$$\forall i, y_i(w^T \phi(s_i) + b) \geq 1 - \xi_i \quad \forall i.$$

Thus, the kernel allows a nonlinear relationship among the features, while keeping the computations in the original input space. An example of a kernel is the polynomial kernel $K(x, y) = (x^T y + c)^d$; if $d = 1$, this is the linear kernel.

3.3.2. SVMs for multiple classes

While support vector machines were initially developed for binary classification, a multi-class extension has been implemented using pairwise comparison. This software is available at <http://www.csie.ntu.edu.tw/cjlin/libsvm>; LIBSVM allows the user to specify the kernel type as well as several other parameters.

3.4. Maximum entropy

The maximum entropy (maxent) approach is rooted in information theory and has been successfully applied to many fields including physics and natural language processing. It creates a model that best accounts for the available data but with a constraint that without any additional information the model should maximize entropy. In other words, the model prefers a uniform distribution by maximizing the conditional entropy. The maximum entropy model was originally developed by Berger et al. (1996) for natural language applications such as information retrieval and speech recognition. Jeon and Manmatha (2004) adapted the model for images. We follow the derivation of Jeon and Manmatha (2004) applying maxent to the problem of shape classification.

Given pairs of training data, $(\vec{s}_1, y_1), (\vec{s}_1, y_1), \dots, (\vec{s}_n, y_n)$ where \vec{s}_i represents the image and y_i is the class label we need to create feature functions or predicates as follows. In order to apply the discrete maximum entropy model to classify the images we discretized the feature vector \vec{s} to get K discrete tokens, s_k . Maxent requires the creation of predicates, $f(s_k, y_i)$, where the value of the predicate is 1 if the discretized tokens for the image contain s_k and the image has the class label y_i and 0 otherwise.

The expected value of the predicate with respect to the available training data is given by Eq. (8) where $\tilde{p}(s_k, y_i)$ is the empirical probability distribution that is calculated from the training data. Setting the expected value for the predicate with respect to the estimated model being equal to the expected value obtained from the training data we obtain Eq. (9).

$$\tilde{p}(f) = \sum_{s_k, y_i} \tilde{p}(s_k, y_i) f(s_k, y_i), \quad (8)$$

$$\sum_{s_k, y_i} \tilde{p}(s_k, y_i) f(s_k, y_i) = \sum_{s_k, y_i} \tilde{p}(s_k) p(y_i | s_k) f(s_k, y_i). \quad (9)$$

Since, there are many possible distributions that could satisfy this relationship, using the idea behind the maximum entropy model that maximizes conditional entropy is chosen which is done using Lagrange multipliers for each predicate f_j as in Berger et al. (1996) in Eqs. (10) and (11).

$$H(p) = - \sum_{s_k, y_i} \tilde{p}(s_k) p(y_i | s_k) \log p(y_i | s_k), \quad (10)$$

$$\Lambda(p, \lambda) = H(p) + \sum_j \lambda_j (p(f_j) - \tilde{p}(f_j)). \quad (11)$$

Fixing λ we can obtain a solution by maximizing p (Eqs. (12) and (13)) where $Z(s_k)$ is normalization factor obtained by setting $\sum p(y_i | s_k) = 1$

$$p(y_i | s_k) = \frac{1}{Z(s_k)} \exp \left[\sum_i \lambda_i f_i(s_k, y_i) \right], \quad (12)$$

$$\Psi(\lambda) = \sum_{s_k} \tilde{p}(s_k) \log Z(s_k) + \sum_i \lambda_i \tilde{p}(f_i). \quad (13)$$

We use Zhang Le's maximum entropy toolkit as the implementations for our experiments [<http://www.nlplab.cn/zhangle>].

3.5. Cross-media relevance model

The Cross-Media Relevance Model (CMRM) was developed by Jeon et al. (2003) and is an extension of relevance-based language models for the task of image annotation. Relevance models were developed so as to perform query expansion for information retrieval in a more formal way and have been used successfully for many information retrieval tasks such as cross-language information retrieval (Lavrenko et al., 2002). Adapting this idea to the field of multimedia information retrieval (Jeon et al., 2003) created Cross-Media Relevance Model (CMRM) for automatically annotating and retrieving images.

3.5.1. Annotation

Given a training set of images T with annotations c_i (class labels), the CMRM allows one to learn a joint probabilistic model which may then be used to annotate test images. The features are assumed to be discretized. We assume that for each test image I there exists an underlying probability distribution $P(\cdot|I)$, the relevance model for I . We assume that the class label and the features are random i.i.d. samples from this relevance model. For classification we compute the probability of $P(c_i|I) = P(c_i|s_1, \dots, s_k)$ where the s_k are discretized features obtained from I (Jeon et al., 2003). This conditional probability may be easily computed if we know the joint probability $P(c_i, s_1, \dots, s_k)$. Assuming independence between the annotation and the feature vector, the joint probability may be estimated as

$$P(c_i, s_1, \dots, s_k) = \sum_{J \in T} P(J) P(c_i|J) P(s_1, \dots, s_k|J), \quad (14)$$

where J is a training image and the sum is over all training images J . The above equation may also be interpreted as a mixture over all training images. Assuming the features are independent we can write this as

$$\begin{aligned} P(c_i, s_1, \dots, s_k) &= \sum_{J \in T} P(J) P(c_i|J) \prod P(s_1, \dots, s_k|J) \\ &= \sum_{J \in T} P(J) P(c_i|J) \prod P(s_k|J). \end{aligned} \quad (15)$$

We assume uniform priors and obtain $P(c_i|J)$ and $P(\vec{s}|J)$ using smoothed maximum likelihood estimates,

$$P(c_i|J) = (1 - \alpha) \frac{\#(c_i, J)}{|c_J|} + \alpha_J \frac{\#(c_i, T)}{|c_T|}, \quad (16)$$

$$P(s_k|J) = (1 - \beta) \frac{\#(s_k, J)}{|s_J|} + \beta_J \frac{\#(s_k, T)}{|c_T|}, \quad (17)$$

where $\#(c_i, J)$ is 0 or 1 depending on whether c_i is the annotation for image J , c_J is the number of class labels assigned to image J , and c_T is the total number of class

labels assigned to all the images in T . $\#(s_k, J)$ is 0 or 1 depending on whether s_k describes image J , s_J is the number of discretized features in J , and s_T is the total count of discretized features assigned to all images in T . $\#(c_i, T)$ and $\#(s_k, T)$ are the counts of the class label and features in the training set. α and β are smoothing parameters estimated from the data. Each test image is then annotated with a vector of probabilities for all the words c_i in the vocabulary. In other words each image is annotated with the probability that it is generated from each of the given classes. A given image belongs to a particular class c_i if it is most likely that it has been generated from the class c_i . CMRM allows for multiple annotations for each image. This means if an image has more than one class label (e.g., n class labels) the model will be able to find top n class labels for an image. This relaxes the restriction present in most machine learning classification algorithm that an instance must belong to only one class. The approach for the multi-class assignment problem is similar to the single class problem where a model is learned for each label from the training set. During the testing process the top n labels are chosen to annotate each test image.

3.5.2. Retrieval

We can also perform retrieval with multi-word text queries by using a language model approach (Lavrenko et al., 2002) by computing the probability of generating the query given the image

$$P(Q|I) = \prod_{m=1}^{|Q|} P(c_m|I). \quad (18)$$

Here, c_m corresponds to each of the words in the query. Using the annotation model this becomes

$$P(Q|I) = \prod_{m=1}^{|Q|} \sum_{J \in T} P(c_j|J) \prod_{k=1}^K P(s_k|J) P(J). \quad (19)$$

In the case with only single word queries, since all the images belong to only one class, given a text query which is a class label, this implies that retrieval is done by simply ranking the images according to the annotation probability of the query word.

4. Features

Good classification models are not sufficient to appropriately classify and retrieve images but instead have to work in conjunction with good features that suitably characterize the images. In the case of shape-related images it is frequently desired that the features be invariant to rotation, translation, and change in scale. Latecki et al. (2000) compared a number of different shape-related features that would possess the above-mentioned characteristics. The broadly defined shape descriptors are classified into two categories:

1. *Contour-based descriptors*: The contour of an object is obtained and represented appropriately which serves as a shape descriptor.
2. *Image-based descriptors*: The shape descriptor is obtained by summing the pixel values in the image and deriving a number of parameters from it.

The features described in this paper fall into the first category. In the following sections we will describe how the features were extracted including the preprocessing involved. Following the description of the representation used to characterize the contour of the object we will present a set of operations performed to ensure that the shape descriptors are invariant to translation, rotation, and change in scale. The paper describes two different sets of features to characterize the image where both sets of features are derived from the contour of an image.

4.1. Preprocessing

In order to extract features we assume that each object can be characterized by a single closed contour. To ensure this we preprocess each image by filling in all the holes and breaks in the contour of the object.

4.2. Centroid distance function

Our first feature set involved extracting a single feature vector from each object using a centroid distance function ([Zhang and Lu, 2003](#)). We calculate the distance between every point on the boundary of the object and the centroid of the object. The feature results from recording one number per pixel in the contour, thus creating a time series of the width of the length of the contour. The centroid distance function is given below:

$$s(t) = \left[(x(t) - x_0)^2 + (y(t) - y_0)^2 \right]^{1/2}, \quad (20)$$

where x_0 and y_0 are the centroids in the x and y directions, respectively. The centroid distance function provides us with a unique shape signature for each object.

4.3. Profile feature set

This feature set representation of the shape of the object is based on a modification of the features described by [Rath and Manmatha \(2003\)](#) for the purpose of handwriting recognition and retrieval. The features are:

1. *Horizontal projection profile*: The value of each column is the number of white pixels in the column .
2. *Vertical projection profile*: The value of each row is the number of white pixels in the row.
3. *Upper shape profile*: The value of each column is the distance between the top of the image and the first white pixel in the column.
4. *Lower shape profile*: Same as the upper word profile except that it is the distance from the bottom of the image.
5. *Right shape profile*: Distance from the right of the image.

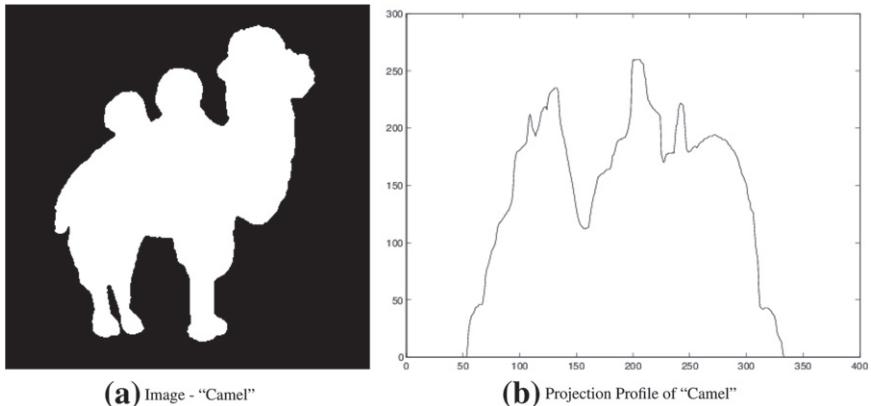


Fig. 2. Preprocessed image of a camel and its projection profile.

6. *Left shape profile*: Distance from the left of the image.

An example of the projection profile for an image labeled “camel” can be seen in Fig. 2.

4.4. Fourier transform

Both the centroid distance function and the profile features capture the shape of an object in great detail. However, the length of the feature vector varies depending on the object and our models for classification and image retrieval require a fixed length feature vector. In order to convert each of the aforementioned profiles into a fixed length feature vector we compute its Discrete Fourier Transform.

We perform the Discrete Fourier Transform on the time series, $s = s_0, s_1, \dots, s_{n-1}$ where $s_k = s(t)$. This results in a frequency space representation $S = S_0, S_1, \dots, S_{n-1}$ for $0 \leq k \leq n - 1$

$$S_k = \sum_{t=0}^{n-1} s_t e^{-2\pi i k / n}. \quad (21)$$

The DFT takes into account that different objects are of different sizes since a single period of the DFT basis function is equal to the number of sample points. The lower order coefficients provide an overall shape representation while the higher order coefficients provide the details of the shape.

From the DFT of the centroid distance function we extracted the first 40 coefficients and used them as features. These lower order coefficients are sufficient features since they are a good approximation of the complete frequency space representation and the goal is not to represent the original shape in great detail but rather to get a good approximation of it with a fixed number of feature values.

In the case of the profiles features, each profile is a one-dimensional curve. We computed the DFT of each profile and extracted its first seven coefficients. Therefore, $6 * 7 = 42$ features are obtained from the profile. We do not consider higher order

coefficients since they frequently contain information about noise. Combined with the 40 features from the centroid distance function this gives a total of 82 features which are used to describe the shape of each object.

4.5. Invariance

Shapes generated through scaling, rotation, translation of a shape are similar to the original shape and the shape descriptors chosen should be invariant to these operations (Zhang and Lu, 2003). It is possible to adjust the Fourier descriptors such that they are invariant to changes in scaling, rotation, translation, and a change in starting point of the contour. Let $s(t)$ be the original shape signature and $S_0, S_1, \dots, S_n - 1$ be its corresponding Fourier representation. We note that we only use the magnitude of the Fourier descriptors. The operations on shapes and the resulting Fourier coefficients can be described as follows:

1. *Change in starting point*: The change in starting point can be expressed as

$$s'(t) = s(t) + s(t + \tau), \quad (22)$$

where $s(t)$ describes the original contour. The new Fourier coefficient may be expressed as

$$S'_k = e^{int} S_k, \quad (23)$$

where S_k is the k th Fourier coefficient of the original shape. The magnitude of the Fourier coefficient is, therefore, invariant under a change in starting point.

2. *Rotation*: Given that the center of object is at the origin, rotation of a curve $u^{(o)}(t)$ about the origin by an angle θ results in a new curve

$$s'(t) = s(t)e^{i\theta}. \quad (24)$$

The Fourier coefficients of this new contour may be expressed as

$$S'_k = S_k e^{i\theta}. \quad (25)$$

Again, the magnitude of the Fourier coefficient is invariant under rotation.

3. *Scaling*: The scaling of a curve can be expressed as

$$s'(t) = h \cdot s(t), \quad (26)$$

where h is the scaling constant. This results in the new Fourier coefficient

$$S'_k = h \cdot S_k. \quad (27)$$

If we rescale the Fourier coefficients so that one of the coefficients is always one, then the Fourier coefficients are also invariant under scaling.

4. *Translation*: The translation of a shape may be expressed as

$$s'(t) = s(t) + c, \quad (28)$$

where c is a constant by which the shape is translated. This results in new Fourier coefficients that can be expressed as follows:

$$S'_k = \begin{cases} S_k + c & \text{if } n = 0, \\ S_k & \text{otherwise.} \end{cases} \quad (29)$$

Hence, all the Fourier coefficients are invariant to translation except the first coefficient. The first coefficient is not a good descriptor of the shape since it only reflects the position or average scale of the shape (Zhang and Lu, 2003), and so it can be ignored while examining the shape of a contour.

5. Classification experiments

We conducted experiments on the MPEG-7 shape dataset that contains the silhouette of 70 different types of objects (Latecki et al., 2000). Each of the 70 classes contains 20 images. Certain classes, such as “lizard(sic)” and “butterfly,” have considerable within class variance while other classes, for example “bottle” and “tree,” are fairly uniform. Figure 1 demonstrates a sample of images from the database. The dataset is further complicated as some images belonging to different classes resemble each other more than they resemble remaining images from their own class. An example of this can be seen in Fig. 8. The MPEG-7 database was selected for our experiments so as to make our results comparable with prior work on shape-based image retrieval. In our experiments we compared the CMRM Model with standard techniques for classification like Naïve Bayes, Linear Discriminant Analysis, and Support Vector Machines. We ran each of the aforementioned techniques on the database after splitting it into sets with 80% training and 20% test as well as 20% training and 80% test.

In order to apply the Naïve Bayes classifier each of the components of the feature vector representing the normalized Fourier coefficients of the centroid distance function was modeled as a Gaussian and we computed the mean and variance for each element of the vector. We classified each test instance using calculated sufficient statistics. Similarly each component of the vector representing the profile features is modeled as a Gaussian in order for us to apply the Naïve Bayes classifier.

We used the LIBSVM toolkit (<http://www.csie.ntu.edu.tw/cjlin/libsvm>) for SVMs. Using the toolkit we normalized the features using the built-in scaling feature. For both sets of features we experimented with the following kernels where γ , r , and d are kernel parameters:

1. *Linear*: $K(x_i, x_j) = x_i^T x_j$.
2. *Polynomial*: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d$, $\gamma > 0, d > 1$.
3. *Radial basis function*: $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$, $\gamma > 0$.
4. *Sigmoid*: $K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$.

However only the linear kernel performed well. We show the results for the various kernels to demonstrate the differences between them.

For classifying the dataset using LDA we used the R mathematical software. We first scale the features to lie between $[-100, 100]$ and then fed it into the built-in command for Linear Discriminant Analysis. Similar to the other techniques we experimented with different samples of training and test sets.

Table 1
Classification results with 80% training and 20% test

	Centroid distance (%)	Profile (%)
Relevance model	75	79.8
Naïve Bayes	60	67
LDA	70.7	71.3
Maximum entropy	64	68
SVM (kernel)		
Linear	67.87	73
Polynomial ($d = 2$)	34.35	38
Radial	20	21
Sigmoid	20.3	18.4

In applying the CMRM we first had to discretize the features since the CMRM is a discrete model. In order to learn the smoothing parameters we split the training set into training instances and validation instances. Using the training set we learnt the parameters that maximized the correct classification of the validation instances.

The results of all the different techniques are shown in Table 1 where we used 80% of the data for training and Fig. 3 shows some examples of correctly and incorrectly classified instances.

6. Retrieval

Since the relevance model with profile features performed the best in our classification experiments we used this model to perform retrieval experiments using both the MPEG-7 and the COIL-100 databases.

6.1. Retrieval with MPEG-7 database

In order to perform retrieval experiments on the MPEG-7 database we complemented the projection profile and upper/lower profiles by also calculating them for the shape at a 90° rotation angle. Again we discretize the features. We performed retrieval experiments using tenfold cross-validation. For the retrieval experiments, we ran 70 ASCII queries on the testing set. Each of the unique 70 shape category labels serves as a query.

For each cross-validation run we have a 90% training 10% testing split of the entire database. We performed retrieval experiments on the training portion in order to determine the smoothing parameters α and β for the visterm and annotation vocabularies. The smoothing parameters that yielded the best retrieval performance are then used for retrieval on the testing split.

Table 2 shows the mean average precision results we achieved with the 10 cross-validation runs. We get very high retrieval performance at 87% mean average precision. It is important to note that in contrast to the common query-by-content

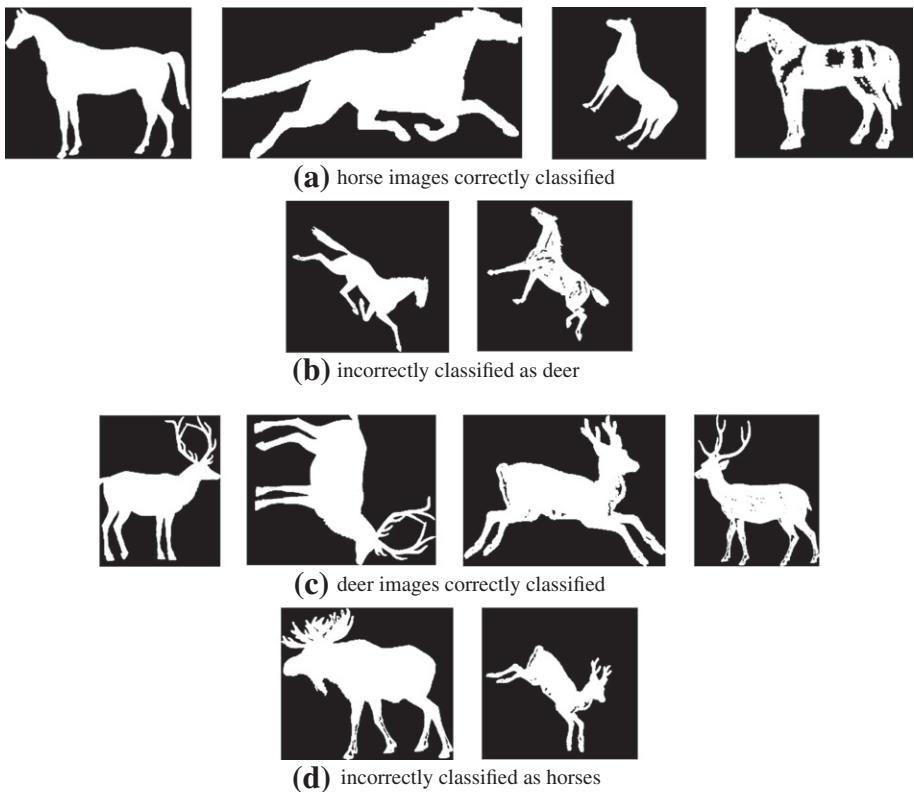


Fig. 3. Sample classification results on MPEG-7 database.

retrieval systems, which require some sort of shape drawing as a query, we have actually learned each shape category concept, and can retrieve similar shapes with an ASCII query. While our results are not directly comparable with the bull's eye evaluation used by Latecki et al. (2000), we note that their results were evaluated over the top 40 ranks while the average precision is computed over all ranks.

6.2. Retrieval with COIL-100 database

For increased complexity we turned to the COIL-100 database (Nene et al., 1996). This database contains 7200 color images of 100 household objects and toys. Each object was placed on a turntable and an image was taken for every 5° of rotation,

Table 2
MAP results for the retrieval on MPEG-7 database

Mean average precision (%)	Standard deviation (%)
87.24	4.24

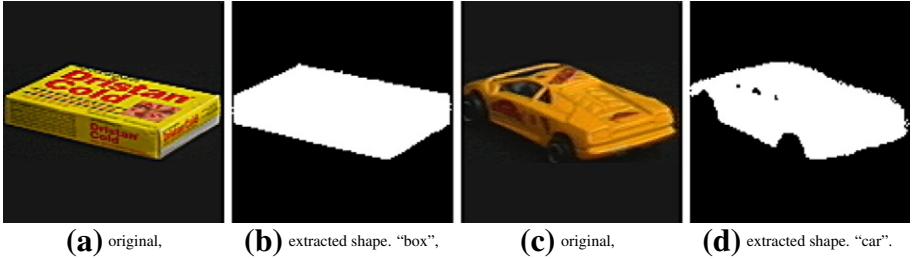


Fig. 4. COIL-100 database examples: original images and extracted shapes with our annotations.

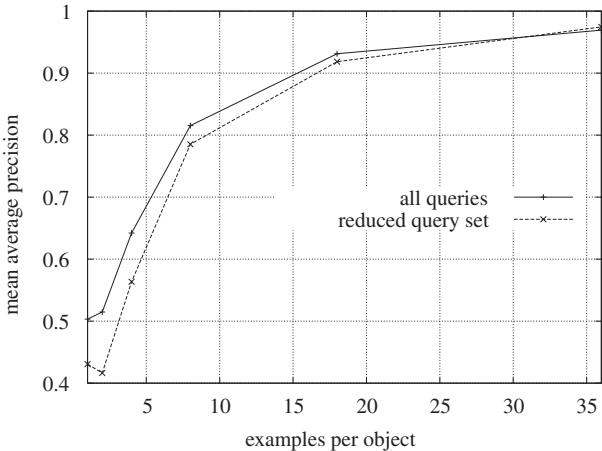


Fig. 5. Retrieval results on the COIL-100 database for different numbers of examples per object. Reduced query set excludes queries for objects that appear invariant under the rotation performed during the database acquisition.

resulting in 72 images per object. We converted the color images into shapes by binarizing the images and normalizing their sizes. Figure 4 shows examples. Note the small difference in the shape of the two objects. Throwing away all the intensity and color information makes the problem more challenging. Our intention is to demonstrate that shape alone provides valuable information.

In order to facilitate retrieval using text queries, each object was labeled with one of 45 class labels (these are also used as queries). The reduced number of classes results from the fact that given the binary image two boxes cannot be distinguished and, therefore, have to be collapsed together into the same class.

After discretizing the features as before, we performed retrieval experiments with varying numbers of training examples per object category. The number of examples per object are (evenly spaced throughout 360° of rotation): 1, 2, 4, 8, 18, and 36. Once the training examples are selected, we pick nine shapes per object at random from the remaining shapes. This set, which contains a total of $9 \times 100 = 900$ shapes, is used to train the smoothing parameters of the retrieval model. From the remaining

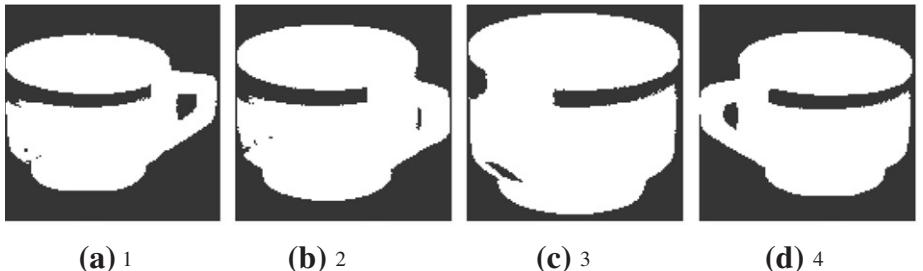


Fig. 6. Ranked retrieval results for the COIL-100 dataset with query, cup.

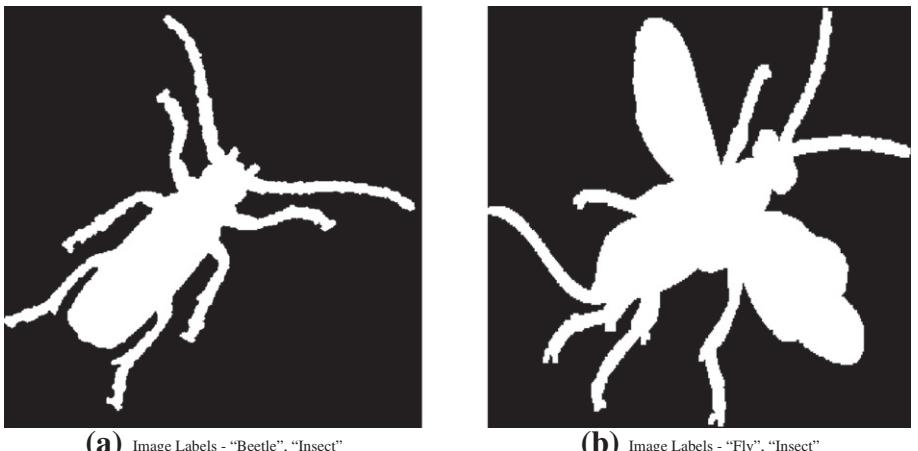


Fig. 7. Both images share a common label, “Insect.”

shapes, another nine shapes per object are selected at random to form the testing set on which we determine the retrieval performance.

Figure 5 shows the mean average precision results obtained in this experiment (“all queries” plot). A retrieval example is shown in Fig. 6.

These results are very encouraging, since they indicate we can perform satisfactory retrieval at around 80% mean average precision for eight examples per object (45° apart) and high performance retrieval at 97% for 36 examples per object (10° apart). Note that this is done exclusively on shape images (without using any intensity information). If other information and a more specialized feature set were used, even higher precision scores could be achieved.

7. Multiple class labels

CMRM is equipped to handle images with multiple class labels. Most prior work related to shape classification and retrieval has focused on images with single class labels. We wanted to examine how well CMRM would perform given that each image

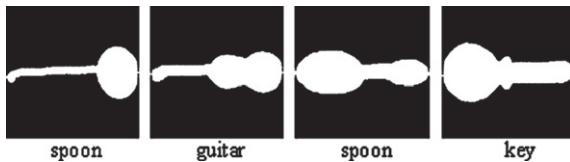


Fig. 8. Instances of class Spoon that are more like instances of another class than each other.

has multiple-labels. Since the MPEG-7 only has 1 label per class we attach an extra label to each of the images that is based on the category of the label of the image. For example, all the images labeled “camel” now also have the label, “mammal” associated with them whereas the images labeled “apple” now also have the label, “fruit.” Since the new label for each image is based on the category of the first label, many images have the same new class label. For instance, the images in Fig. 7 share a common label. In this manner we added 18 labels to the dataset such that now each image may be assigned to 2 out of 88 possible labels. Using CMRM we annotate each test image with the top two annotations describing the image from the entire set of annotations. The classification rate is based on the number of class labels for each image that are predicted correctly. Using this approach we obtain a 86.25% classification rate. This demonstrates the capabilities of the Cross-Media Relevance Model.

8. Summary and conclusions

We have demonstrated the use of different machine learning techniques and the relevance model along with shape features to learn the shape of an object. In particular we compute the Fourier coefficients of two contour-based shape descriptors, the centroid distance function, and the profile features. We use these features with Naïve Bayes, SVMs, and the CMRM to classify the images in the MPEG-7 dataset. Classifying images in this dataset is a hard problem because frequently images of different classes look more like each other than they do to members of their own class as can be seen since in Fig. 8. Since the relevance model with the profile features performs better than the remaining techniques we use the relevance model to retrieve images given ASCII queries corresponding to class labels from the MPEG-7 and COIL-100 databases. The results indicate that our system is competitive with prior work.

Future work includes classifying and retrieving from a larger collection of images and incorporating shape along with other features such as color, texture. We also plan further experiments with multi-label classification that can be useful in many domains.

References

- Adamek, T., OConnor N., 2003. Efficient contour-based shape representation and matching. In: Fifth ACM SIGMM International Workshop on Multimedia, Information Retrieval.
- Barnard, K., Duygulu, P., de Freitas, N., Forsyth, D., Blei, D., Jordan, M.I., 2003. Matching words and pictures. *J. Mach. Learn. Res.* 3, 1107–1135.

- Belongie, S., Malik, J., Puzicha, J., 2002. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (24).
- Blei, D.M., Jordan, M.I., 2003. Modeling annotated data. In: Proceedings of the 26th Annual International ACM SIGIR Conference, Toronto, Canada, July 28–August 1, pp. 127–134.
- Boser, B., Guyon, I., Vapnik, V., 1992. A training algorithm for optimal margin classifiers. In: Proceedings for the Fifth Annual Workshop on Computational Learning Theory.
- Berger, A., Della Pietra, S., Della Pietra, V., 1996. A maximum entropy approach to natural language processing. *Comput. Ling.* 22 (1), 39–71.
- Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification, second ed. Wiley Interscience.
- Jain, A.K., Vailaya, A., 1998. Shape-based retrieval: a case study with trademark image databases. *Pattern Recogn.* 31 (9), 1369–1390.
- Jeon, J., Lavrenko, V., Manmatha, R., 2003. Automatic image annotation and retrieval using cross-media relevance models. In: Proceedings of SIGIR Conference.
- Jeon, J., Manmatha, R., 2004. Using maximum entropy for automatic image annotation. In: Proceedings of International Conference on Image and Video Retrieval.
- Latecki, L., Lakämper, R., Eckhardt, U., 2000. Shape descriptors for non-rigid shapes with a single closed contour. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Lavrenko, V., Choquette, M., Croft, W.B., 2002. Cross-lingual relevance models. In: Proceedings of the 25th Annual International SIGIR Conference, Tampere, Finland, August 11–15, pp. 175–182.
- Mohanty, N., Rath, T.M., Lee, A., Manmatha, R., 2005. Learning shapes for image classification and retrieval. In: International Conference on Image and Video Retrieval, CIVR, Singapore, July 20–22.
- Mokhtarian, F., Abbasi, S., Kittler, J., 1997. Efficient and robust retrieval by shape content through curvature scale space, in: Smeulders, A.W.M., Jain, R. (Eds.), *Image Databases and Multi-Media Search*. World Scientific Publishing, Singapore, pp. 51–58.
- Nene, S.A., Nayar, S.K., Murase, H., 1996. Columbia object image library (COIL-100). Technical Report CUCS-006-96, February.
- Rath, T.M., Manmatha, R., 2003. Word image matching using dynamic time warping. *Proc. Conf. Comput. Vis. Pattern Recog.* 2, 521–527.
- Super, B., 2004. Learning chance probability functions for shape retrieval or classification. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Veltkamp, R.C., Hagedoorn, M., 1999. State-of-the-art in shape matching. Technical Report UU-CS-1999-27, Utrecht University, The Netherlands.
- Zhang, D., Lu, G., 2003. Evaluation of MPEG-7 shape descriptors against other shape descriptors. *Multimed. Syst.* 9 (1).

This page is intentionally left blank

Visual Search: A Large-Scale Perspective

*Robinson Piramuthu, Anurag Bhardwaj, Wei Di,
and Neel Sundaresan*

eBay Research Labs, San Jose, CA, USA

Abstract

Advances in the proliferation of media devices as well as Internet technologies have generated massive image data sets and made them easier to access and share today. These large-scale data sources not only provide rich test beds for solving existing computer vision problems, but also pose a unique challenge for large-scale data processing that demands an effective information retrieval system to browse and search. This is also motivated by many real-world applications where visual search has been shown to offer compelling interfaces and functionalities by capturing visual attributes better than modalities such as audio, text, etc. In this chapter, we describe state-of-the-art techniques in large-scale visual search. Specifically, we outline each phase in a typical retrieval pipeline including information extraction, representation, indexing, and matching, and we focus on practical issues such as memory footprint and speed while dealing with large data sets. We tabulate several public data sets commonly used as benchmarks, along with their summary. The scope of data reveals a wide variety of potential applications for a vision-based retrieval system. Finally, we address several promising research directions by introducing some other core components that serve to improve the current retrieval system.

Keywords: content-based image retrieval (CBIR), computer vision, large scale, features, hash

1. Introduction

1.1. Data everywhere

Advancement in technology such as high speed networks, powerful computing devices, storage devices, portable devices, and display technology has greatly increased the availability/accessibility of information which promotes fast and easy sharing. This information exists in diverse forms such as text, audio, video, and

covers a wide variety of dimensions such as temporal, location, user information (demographics, personal preference, connections to other users), product reviews. Here are a few examples to give perspective to the growing size of data. The number of users of Facebook has increased from a million in 2004 to over a billion in 2012, with over 300M photos uploaded *every day*. Over 72 h of video are uploaded to YouTube *every minute*. Just on Facebook, over 500 years worth of YouTube videos are viewed every day. Over 1.8M pictures are uploaded on Flickr *every day*. Taking e-commerce as an example, over 15M items are listed on eBay *every day*, with each containing at least one image, along with rich meta data such as title, description, price. This explosion of data has created the need for information retrieval systems to browse and search these collections. Only considering eBay, a pair of shoes is sold every 3 s, a women's dress is sold every 4 s, a passenger vehicle is sold every 2 min, a boat is sold every 19 min, the advantages of providing customers with powerful visual search tools can be significant.

1.2. Scope

Of all the forms of large-scale data, visual data is of particular interest for multiple reasons: (i) Such data sources provide novel interaction mechanisms that are not possible in other sources such as text and audio; (ii) They are ubiquitous across languages and geographies (no translation required) which makes them appealing for various real-world applications; and (iii) Visual media in general are a more potent communication tool as opposed to other forms. In this chapter, we will focus on visual data. In particular, we will discuss images and most of what we discuss is extensible to videos. Sources of visual data include personal photo albums, movies, TV programs, news, sports, security and surveillance, medical applications, biometrics, e-commerce, social networking. The purpose of query can be informational (landmark, museum, maps), educational (leaves, breeds), medical diagnosis, media search (DVD covers, magazine, movie posters), e-commerce (fashion, wine labels, catalogs), forensics, identity (bio-metrics), security (video surveillance). We will present techniques for retrieval of images based on visual content only. However, we assume that images in the collection may be structured or labeled based on meta data such as title, keywords, topics that are associated with them. Such a retrieval system that is driven by visual information is referred to as Content Based Image Retrieval (CBIR), which will be the scope of our discussion.

1.3. Outline

We start with the importance of large data in Section 2. One of the core parts of CBIR is how we extract and represent visual information. Most common cues include color, texture, shape, depth (vanishing points, occlusion), illumination and shading, grouping (similar color, texture, proximity, common fate), parts/attributes, 3D geometry, geometric layout, scene understanding, activity understanding, tracking, and causality. It is important for extracted information to be robust to variations in view points, lighting conditions (Fig.2), non-rigid deformations, occlusion, scale, blur, motion, background clutter, and intra-class variations (Fig.1). Extraction of a majority of these cues is described in Section 3. The next important part of CBIR is to



Fig. 1. Illustration of large variations within a single class “chair.” Humans can easily group these as chairs even though they might not have seen these examples before.



Fig. 2. Images from yale faces B (Georghiades et al., 2001) showing several variations of the same face due to lighting conditions and pose.

match the extracted information from a query with that in the collection/database. This is addressed in Section 4. With large data, precision and recall are not the only important factors. Having the infrastructure to extract information and match it with huge data collections is not sufficient. There are other practical issues involving memory and speed constraints. Section 5 shows how this is tackled. In order to quantitatively evaluate the effectiveness of the CBIR system, well-defined benchmarks are required. Several commonly used data sets are summarized in Section 6. This also gives an idea of CBIR applied to different use-cases. Finally, in Section 7 we talk about other items that are essential for large-scale retrieval, which are not discussed in detail in this chapter.

2. When is big data important?

Unlike many problems in physics, problems that deal with inference by humans are hard to model elegantly and still work well practically (Halevy et al., 2009). Large data comes to the rescue in such cases. For example, we can teach the computer to identify a bird species by providing it a huge number of exemplars for training. In this section, we will motivate why we need large data for many interesting problems in computer vision.

First we illustrate how the underlying distribution of data dictates the size of training data for classification and retrieval models. We look at how much data is required to reasonably estimate shape parameters for normal distribution and long tailed distribution. Let us assume that the shape parameter is estimated based on n samples. Consider these n samples to be i.i.d. and drawn from normal distribution $\mathcal{N}(\mu, \sigma^2)$. The joint probability distribution for these samples is given by

$$\begin{aligned} f(x_1, \dots, x_n | \mu, \sigma^2) &= \prod_{i=1}^n f(x_i | \mu, \sigma^2) \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2} \cdot \sum_{i=1}^n (x_i - \mu)^2\right). \end{aligned} \quad (1)$$

The expected value of maximum likelihood estimate of variance σ^2 (the shape parameter) is given by

$$E[\hat{\sigma}^2] = \sigma^2 - \frac{\sigma^2}{n}. \quad (2)$$

Note that this is a biased estimate and becomes unbiased as $n \rightarrow \infty$.

Now consider each x_i to be drawn from a long tail distribution such as Pareto distribution given by

$$f(x_i | \alpha) = \begin{cases} \alpha \left(\frac{x_{\min}^\alpha}{x_i^{\alpha+1}} \right), & \text{for } x_i \geq x_{\min}, \\ 0, & \text{for } x_i < x_{\min}. \end{cases} \quad (3)$$

Long tail distributions have data values more spread out in the tail than normal distribution. The expected error in maximum likelihood estimate of shape parameter α is given by Newman (2005)

$$\sigma = \frac{\hat{\alpha}}{\sqrt{n}}, \quad (4)$$

which approaches 0 as $n \rightarrow \infty$. Compare this with (2) where the error decays at a faster rate of $1/n$ instead of $1/\sqrt{n}$. This means that we need many more points to model long tail distribution such as Pareto accurately, than modeling normal distribution.

When dealing with large amounts of categorical data (such as words in dictionary) or distribution of integrated quantities (such as an area of connected components, count of patterns of pixels (Caron et al., 2007)), it has been empirically observed that they follow Zipf's law (Newman, 2005). Zipf's law is a discrete power law probability distribution, that is long tailed. Pareto distribution can be considered as



Fig. 3. Example showing different renditions of the same product (Nintendo DS Lite) with similar pose. The first two images and the last two images have similar pose. Note the difference in quality of pictures taken by different cameras as well as the variations in picture composition.

the continuous distribution counterpart of Zipf's law. Such problems that deal with long tail distribution require more samples for accurate modeling.

Second, many computer vision problems also suffer from inherent variability in data. Examples are lighting conditions (Fig. 2), viewpoint, pose (Fig. 3), occlusion, noise, blur, and camera specifications (mobile camera vs SLR). Due to these variations, a large number of exemplars are required to represent an object or semantic entity such as scene.

Third, there have been several advances toward making the extracted features to be invariant to many of these factors that introduce variability. Part of the strategy to achieve this is to concatenate features at different scales and orientations. This results in large feature vectors that lead to the curse of dimensionality (Bellman, 1957), which in turn demands an increase in the size of training data.

Besides the variations mentioned above, many practical problems have huge intra-class variations, which complicate the modeling step. For example, shoes, clothing, and furniture come in various styles and birds, flowers, and animals come in various species. See Fig. 1 for various instances of chair. Enough representative samples from these fine-grained categories need to be included in the model for better recall.

3. Information extraction and representation

Throughout this chapter, when we talk about an image, we mean a digital image that is a rectangular array of pixels, the values of which represent the average color in a square area around the location of pixel. Each pixel could hold a scalar value representing intensity or a triplet representing color values (typically corresponding to red, green, blue channels). Most results also extend to hyper spectral images that have more than three channels. We use the term "feature" or "feature vector" or "descriptor" to denote the information extracted from the image.

Before we talk about details of information extraction, let us take a brief look at how visual information is processed by the human brain. The cerebral cortex is the thin outer layer neural tissue of the cerebrum (forebrain). The visual cortex of the brain is that part of the cerebral cortex that is responsible for processing visual information. The visual cortex is divided into several parts namely V1 (a.k.a. primary visual cortex), V2, V3, V4, V5, and a few more. Light enters the eye and is captured on the retina. This triggers nerve impulses which are transmitted to the brain through optic nerves. The lateral Geniculate Nucleus (LGN) receives the signal from optic

nerves and transmits it to the visual cortex. It takes about 20–30 min for the retina to send nerve impulses. After 30–50 min since light entered the eye, LGN starts to receive the signal (Nowak and Bullier, 1997). V1 gets the signal during the window of 40–60 min, V2 in 50–70 min, and V4 in 60–80 min. Color and brightness information are picked up by photoreceptors in the retina. With the current understanding, V1 can discriminate small changes in orientations, spatial frequencies, color and can understand simple visual forms such as edges and corners. V2 can understand size and weak shape such as illusory contours. V3 processes global motion and is also the most sensitive to depth perception. V4 can understand basic geometric shapes but not complex shapes like faces. There are other areas that specialize in faces, buildings and scenes, and common objects. Why is this important? This shows the priority/hierarchy in which a visual signal is processed. This suggests that colors are processed first, then orientation, size, contours, motion, depth, geometric shape, and complex shapes. Using such a scheme to extract information would be a bottom-up approach.

There is another philosophy which follows exactly the opposite approach. Max Wertheimer, a Gestalt¹ psychologist from the early 1900s once quoted (Wertheimer, 1923): “*I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of color. Do I have ‘327’? No. I have sky, house, and trees.*” Gestalt psychologists believe that the brain perceives the scene as a whole, before decomposing it into parts such as lines and curves. Max Wertheimer introduced the Gestalt principles of grouping in 1923 based on proximity, similarity, closure, common fate, symmetry, continuity, good gestalt, and past experience. Information could be extracted based on these principles on how objects are grouped in the given image. The choice of approach to use really depends on the use-case. In fact, there are approaches that combine both top-down and bottom-down schemes (Zelinsky et al., 2005; Kokkinos et al., 2006; Levin and Weiss, 2009). It should be noted that low-level image features can be generated based on the Gestalt dimensions for grouping (Fig. 4).

In this section, we will show how to extract information based on color, texture (low-level abstraction of spatial distribution), and shape. The goal is to extract a vector of numbers from a single image that captures relevant information for the problem at hand. We call this vector a *feature vector*. This single feature vector can be generated either by decomposing the entire image in a different informational space as a whole or by extracting information from subsets of an image and then aggregating their statistics (parametric or non-parametric). We summarize these into three broad methods:

Transform coding: The complete image is decomposed into an informational space. A suitable set of basis functions (orthogonal or not) may be used to decompose the image into their span. These basis functions may be derived independent of data or not. An examples of the former are discrete Fourier transform, discrete cosine transform, Fourier–Mellin transform, geometric moments. An example of the latter is principal component analysis (PCA), where the given image is projected onto the dominant eigen vectors (Turk and Pentland, 1991). Images must be pre-processed appropriately before applying PCA. The most common preprocessing steps include

¹ Gestalt is a word in German meaning the essence or *shape* of an entity’s *complete form*.

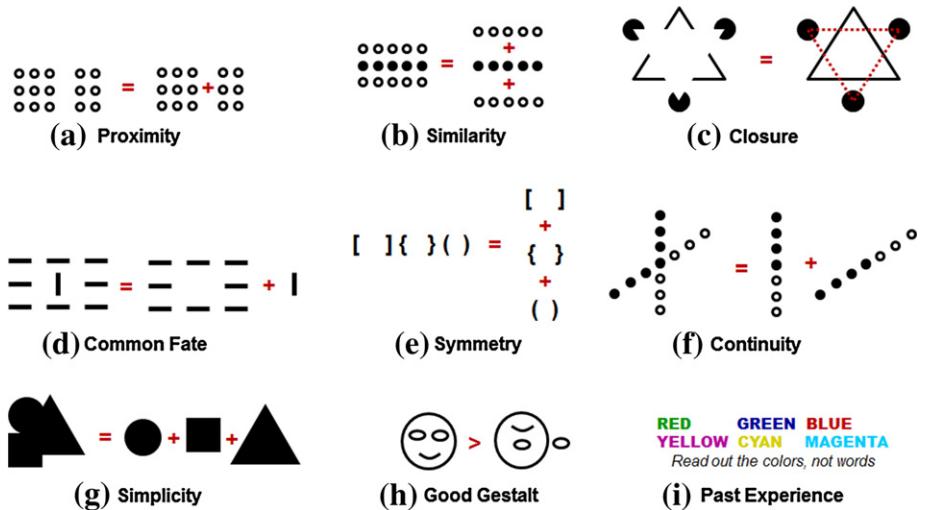


Fig. 4. Grouping based on Gestalt Laws. These laws can be approximated/represented by mathematical expressions and used as extracted information. Annotations in red indicate what is usually perceived. (a) Closer objects look more related. (b) Each row is perceived together. Objects are grouped based on whether they are filled or not. (c) This is the famous Kanizsa triangle illusion. Overlaid white triangle is seen to occlude three black circular disks. Contours of wedges are ignored and seen as occluded circular disks. (d) Orientation is used as fate to group objects. (e) Symmetry of objects is used to group even though the grouped objects are not the closest neighbors. (f) Continuity is used to group rather than similarity. (g) Complex shapes are broken down into simple shapes. (h) Objects that form a pattern when grouped have better Gestalt than those that do not form a pattern. (i) Due to past experience, words are read out more often rather than color. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this book.)

illumination correction and alignment of reference points on the image. In both cases, the projection vector is used as the feature vector. A major drawback of transform coding is its sensitivity to occlusion. This is the reason why transform coding is more popular for scene retrieval and retrieval of rigid objects where alignment is relatively easy. To mitigate the effect of occlusion, the image may be partitioned into blocks, say 4×4 , before applying transform coding to each block. The projection vectors from each block are then concatenated to get the final feature vector.

Direct aggregation of local features: The image is divided into subsets. These subsets can be overlapping or not. The union of these subsets can be smaller than the image (example: only salient parts of image may be used). Some examples of useful subsets are individual pixels, individual pixels along with their local neighborhood, salient points (a.k.a. interest points) in the image along with their local neighborhood, uniform rectangular patches sampled at regular intervals, rectangular patches sampled from random locations, regions of image with similar properties (a.k.a. region segmentation) such as color, salient regions along with their complement, several objects in the image (a.k.a. object segmentation). The same set of quantities such as average color, dominant color (Fig. 5), contrast, orientation, convexity, etc. is extracted from each subset. These quantities are then aggregated either parametrically (example: mixture of Gaussians) or non-parametrically (examples: histogram, clustering) to construct a single feature vector for an image.

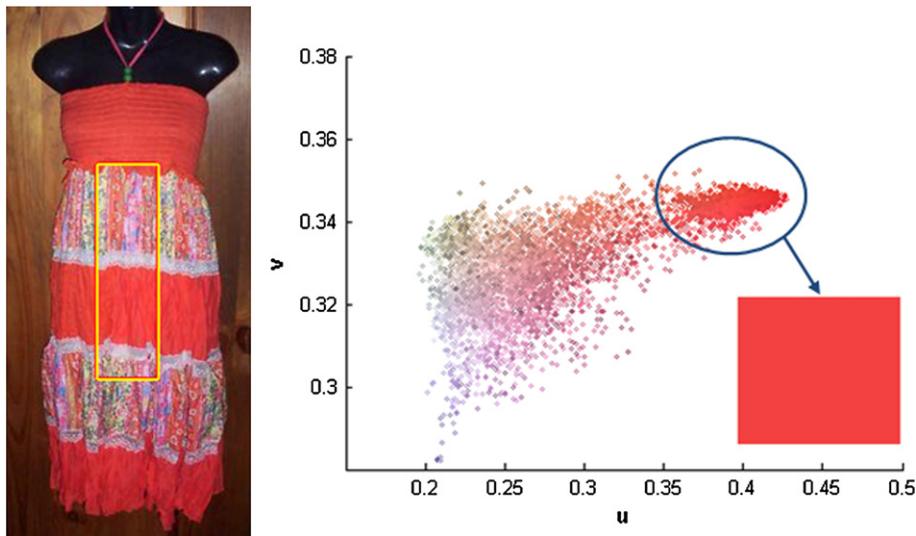


Fig. 5. Color information for the image on left is extracted from the yellow + red rectangle in the center. Dominant color is determined by clustering colors from the sampled region in Luv space (a 3D color space). The scatter plot is shown in u-v space, color coded with true color of pixels. Dominant colors highlight corresponds to the dominant cluster. Top few dominant colors and their statistics may be used as a feature vector for the image. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this book.)

Indirect aggregation of local features: This approach uses an additional step after direct aggregation of local features. In particular, the indirect aggregation is through vector quantization or through clustering by proximity in the feature space, where the feature vectors of subsets lie. Each quantization cell or cluster is represented by a *visual word*. For example, in the case of clothing retrieval, these visual words could mean polka dots, plaid, checkered, solid, etc. The complete set of visual words is called a *dictionary*. The histogram of visual words contained in the image over all visual words in the dictionary is used as the descriptor for each image. Note that spatial information about the occurrence of visual words in the image is lost in this representation. In other words, the descriptor is just a bag of visual words. This approach is often referred to as Bag-of-Words (BoW) (Sivic et al., 2003). This approach is very popular with features discussed in Section 3.2.

We will now present some of the most common ways to extract information from an image.

3.1. Basic information: color, texture, shape

The type of information extracted can be broadly grouped as color, texture, and shape. Below, we summarize how such information can be extracted from an image:

Color: Color features have been commonly used for various image representation tasks. Typically, such features involve converting an image into an appropriate color

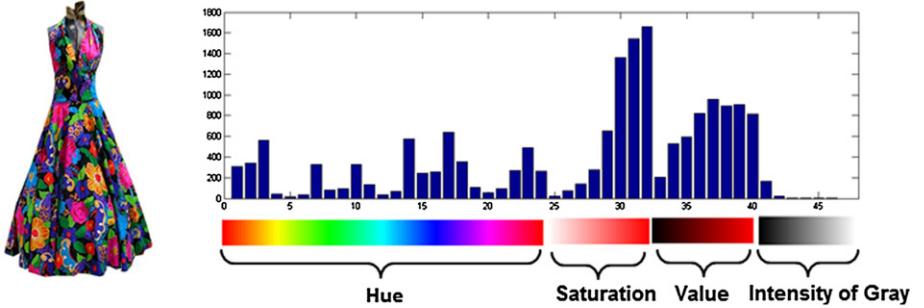


Fig. 6. An example of color histogram. Pixels from the central rectangle were grouped into color and gray, based on the spread of red, green and blue (RGB) values. Pixels with small RGB spread are considered gray. 1D histograms of color channels namely, Hue, Saturation, Value, along with 1D histogram of intensity of gray values were concatenated. The numbers of bins are 24, 8, 8, 8, respectively. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this book.)

space (say, Hue–Saturation–Value) and then binning the pixel values into a histogram of arbitrary number of bins in a uniform or non-uniform grid. One could bin each of the three channels independently in one-dimensional space (see Fig. 6) or together in three-dimensional space or use only the two chromatic channels (Hue, Saturation) in two-dimensional space (see Fig. 5). Note that 2D and 3D representations will be sparser than 1D representation. If we know that only few dominant colors (say 3–5) will be in an image, it is sufficient to use 1D representation. For instance, a typical application may use 24 number of bins for each of Hue, Saturation, and Value channels independently, thus generating a feature vector of length $24 * 3 = 72$ for the image. To effectively match histograms, “cross-talk” between color bins is reduced by using weighted distance-based metrics (Hafner et al., 1995). This technique can be further extended by modeling spatial correlation of color pixels in form of Spatial Chromatic Histogram (SCH) (Cinque et al., 1999). To address varying image illumination in real-world images, color constancy-based algorithms are used (Funt and Finlayson, 1995).

Texture. Texture is typically extracted from the average of RGB channels. It is not uncommon to extract texture features for each color channel and then concatenate them. The increase in dimensionality using this approach is typically not justified by the corresponding improvement in performance. Tamura et al. (1978) described six texture-based features that correspond to human visual perception (see Fig. 7). They identify three primary features that correlate strongly with the human visual perception namely:

- *Coarseness:* This feature models the type of texture present in an image. Images with rough texture (macro texture) tend to have higher coarseness as opposed to images with smooth texture (micro texture) which have lower coarseness. The advantage of computing the coarseness value is to appropriately choose the size of an operator to apply on the image which leads to more robust features.
- *Contrast:* The contrast of an image I is defined as:

$$F_{\text{contrast}} = \frac{\sigma}{\alpha_4^z}, \quad \alpha_4 = \frac{\mu_4}{\sigma^4}, \quad (5)$$

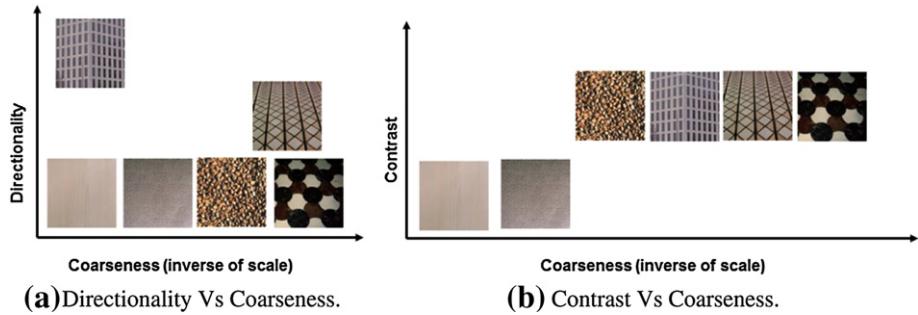


Fig. 7. Dimensions of texture (directionality, coarseness, contrast) as seen by Tamura (Graczyk, 1995; Tamura et al., 1978). Note that coarseness is related to spatial frequency and is inversely proportional to scale in spatial domain.

$$\mu_4 = \frac{1}{xy} \sum_{x=1}^R \sum_{y=1}^C (I(x,y) - \mu)^4, \quad (6)$$

where μ_4 represents fourth moment about the mean μ and variance σ^2 of the gray values of image with R rows, C columns and z is set empirically to 0.25.

- **Directionality:** First, horizontal derivative Δ_H and vertical derivative Δ_V are computed by convolving the image $I(x,y)$ with a 3×3 Sobel operator. For every pixel (x,y) , an angle is calculated as follows:

$$\theta = \frac{\pi}{2} + \tan^{-1} \frac{\Delta_V(x,y)}{\Delta_H(x,y)}. \quad (7)$$

These angle values are further binned into 16 bin histogram H_D which is used as the directionality feature.

One could add other dimensions to texture such as higher order differentials of the intensity of image, stationarity (repetitiveness of pattern), and symmetry. For example, the strength of second-order differential with proper normalization can be used to capture information about depth of focus. We will present few popular methods to capture coarseness, directionality, and contrast.

Wavelet-based features have also received considerable attention for the task of efficient modeling of texture in an image. One such commonly used technique is Gabor wavelets. Manjunath and Ma (1996) in their work describe how to use Gabor wavelet features for browsing and retrieving images with texture data. Gabor features typically consist of response functions from a filter bank containing filters with different orientations and scales. Each filter outputs a response to a certain orientation and scale of texture in the image. By concatenating responses from all such filters, a long vector is obtained, which is further processed to obtain *Gabor* features. The primary visual cortex (V1) is typically modeled using a filter bank of Gabor filters. The mother Gabor wavelet is given by

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp \left[-\frac{1}{2} \left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) + 2\pi j Wx \right], \quad (8)$$

where W = modulation frequency. A class of self-similar functions $g_{mn}(x,y)$ can be generated from the mother Gabor wavelet by dilation and rotation as follows:

$$g_{mn}(x,y) = a^{-m} g(x',y'), \quad (9)$$

where $a > 1$, $m,n = \text{integer}$, $\theta = n\pi/K$, $K = \text{total number of orientations}$, $x' = a^{-m}(x \cos \theta + y \sin \theta)$, $y' = a^{-m}(-x \sin \theta + y \cos \theta)$.

Given an image $I(x,y)$, its Gabor wavelet transform is defined as:

$$T_{mn}(x,y) = \int I(x_1,y_1) g_{mn}^*(x - x_1, y - y_1) dx_1 dy_1, \quad (10)$$

where $*$ represents the complex conjugate. Note that this captures the energy at a specific scale/dilation (related to spatial frequency) and orientation.

Using these equations, the mean value μ_{mn} and standard deviation σ_{mn} of the transform T_{mn} are computed as:

$$\begin{aligned} \mu_{mn} &= \iint |T_{mn}(x,y)| dx dy, \\ \sigma_{mn} &= \sqrt{\iint (|T_{mn}(x,y)| - \mu_{mn})^2 dx dy}. \end{aligned} \quad (11)$$

Finally, texture is extracted as a feature vector generated by concatenating μ_{mn} and σ_{mn} . Note that Gabor feature captures coarseness, directionality, and contrast.

Oliva and Torralba introduce a global descriptor in [Oliva and Torralba \(2006\)](#) called “spatial envelope” that models the “gist” of a scene image. Their work is primarily motivated by the fact that humans can efficiently recognize the gist of a new image in a single glance, irrespective of the complexity of the image. This leads to their observation that the structure of an image can be represented by global image features that are computed over different spatial scales and do not require any computationally expensive task such as segmentation or grouping. Their proposed feature extraction approach is split into two phases:

- *Global feature extraction:* They apply a filter bank of multiscale-oriented filters (similar to Gabor features) and encode the filter output magnitude into an $N \times N$ matrix for each filter. Since these filters are applied at multiple scales K , they downsample each filter output to get exactly $N \times N$ values, thereby obtaining a $N \times N \times K$ dimensional feature vector. They apply PCA (Principal Component Analysis) on this long vector compute weights for each filter. Finally, the weighted combination of filter responses is output as the global features.
- *Spatial envelope:* They define “Spatial Envelope” as a combination of four global scene properties. Properties such as openness and Expansion characterize the properties of boundaries of the space in the image. Properties of content of an image can be characterized by Naturalness and Roughness. These properties are estimated directly from a collection of global features template.

Local Binary Pattern (LBP) was introduced by [Ojala et al. \(1996\)](#) as a measure for local image contrast. For each pixel, all its eight neighbors are chosen and their values are thresholded to a binary value based upon the value of the current pixel. These

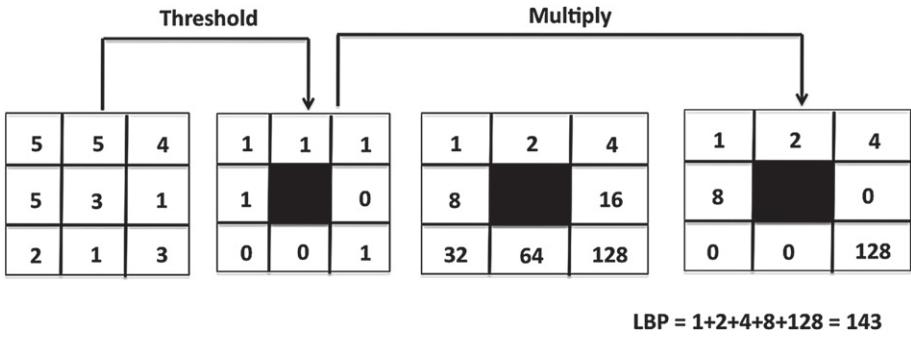


Fig. 8. Figure illustrating the procedure to compute LBP feature, based on an image from Maenpaa (2003).

binary values are then multiplied with corresponding weights (typically in power of two) and summed together to generate the final LBP value for the current pixel. Figure 8 illustrates the procedure to compute LBP features. LBP has the advantage of speed and stability to the effects of illumination.

There are various statistical learning approaches to extract information about texture. Joint distribution of output from filter banks (such as Gabor filter bank) is represented by *textons* that are learned from the data. These textons are representative textures in the data. Texture information is represented by the distribution of these learned textons in the image (Varma and Zisserman, 2005).

Shape: Global shape can be extracted after isolating the object of interest from the background. This process of isolation is called object segmentation. Once object boundaries are identified by segmentation, parametric (example: B-splines) or non-parametric (example: chain code) representation of boundary contour could be used to extract shape information. Shape statistics such as diameter, perimeter, area, eccentricity, circularity, curvature, solidity, bending energy, normalized central moments (Hu, 1962), morphological operators may be used instead of complete contour. We summarize two popular global shape representations: Shape context and HOG.

Shape context was proposed by Belongie et al. (2000) to “solve the correspondence problem between two shapes.” Given an input shape, n points are chosen uniformly to denote the shape context. Formally, for a point p_i on the shape, firstly a histogram h_i of the relative coordinates of rest $n - 1$ points is constructed as:

$$h_i(k) = \#\{q \neq p_i : (q - p_i) \in \text{bin}(k)\}. \quad (12)$$

These bins are constructed in log-polar space to account for sensitivity for nearby points. The cost C_{ij} of matching two points p_i, q_j is measured as:

$$C_{ij} = C(p_i, q_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{[h_i(k) - h_j(k)]^2}{[h_i(k) + h_j(k)]}, \quad (13)$$

where $h_i(k)$ and $h_j(k)$ represent the K -bin normalized histogram at p_i and q_j , respectively.

Global shape descriptors depend heavily on the accuracy of segmentation. They are also sensitive to occlusion. Modern approaches extract shape information without segmenting the image. [Dalal and Triggs \(2005\)](#) proposed Histogram of Oriented Gradients (HOG) features for human detection. Since then, these features have been used extensively for generic image representation tasks as well. These features are motivated by the fact that local intensity gradients can be efficient descriptors for local shape or object appearance. Features are computed by concatenating 1D histograms of edge orientations over various image windows. These responses are also “contrast-normalized” for better illumination invariance.

Several of the features summarized in this subsection capture global information. The primary advantage of using global features is low computation cost for extraction as well as matching. However, in general, global features are sensitive to occlusion, pose and do not capture local representations, especially when dealing with huge number of classes. State-of-the-art features used for complex image retrieval use local features which capture local shape, texture and are invariant to affine transforms and illumination effects. A number of popular local features that have been proposed in the community so far are summarized in the next subsection.

3.2. Local features

Local features describe local image regions. They represent local image blocks into certain feature spaces that efficiently preserve the salient features in the block. Traditionally, such features consist of two important phases: (i) Region Detection—Also referred to as interest point detection. This phase detects all the important regions in the image from which features can be extracted in the subsequent stages and (ii) Feature Extraction—This phase extracts representative features from detected image regions from the previous phase. Traditionally, such features are more discriminant than global features (because of robustness to occlusion and pose variations), but they do suffer from increased computational cost and higher dimensionality of features. A number of popular local image descriptors are described below:

SIFT: Abbreviated as “Scale Invariant Feature Transform,” SIFT was proposed by David Lowe in [Lowe \(2004\)](#). There are two key stages in SIFT namely: (i) key point extraction—The input image is convolved with multi-scale Gaussian Filters. Differences of successive filter responses are computed and key points are extracted as points having maxima/minima of the Difference of Gaussians (DOG) and (ii) key point description—A 16×16 grid is drawn with each key point as its center. This grid is further divided into $16 \times 4 \times 4$ grids. Within each 4×4 grid, gradient orientations are binned in a 8-bin histogram in weights proportional to the gradient magnitude. Finally, 8 bins from all 16 bins are concatenated to generate 128-dimensional key point description.

SURF: [Bay et al. \(2006\)](#) describe “Speeded Up Robust Features” as a faster alternative to SIFT. SURF detector is based on a Hessian Matrix approximation that uses integral image to reduce computation costs. The descriptor uses a distribution of Haar-wavelet output from the neighborhood of detected interest points.

BRIEF: Binary Robust Independent Elementary Features proposed in [Calonder et al. \(2010\)](#) uses binary strings as efficient feature descriptors. Given a patch p of

size $S \times S$, a test τ is defined as:

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) = \begin{cases} 1, & \text{if } I(\mathbf{p}, \mathbf{x}) < I(\mathbf{p}, \mathbf{y}), \\ 0, & \text{otherwise,} \end{cases}$$

where $I(\mathbf{p}, \mathbf{x})$ denotes the pixel intensity of smoothed version of patch \mathbf{p} at $\mathbf{x} = (u, v)^T$. Using this formulation for a set of n_d pairs of location (\mathbf{x}, \mathbf{y}) , the final BRIEF descriptor bit string is represented as:

$$f(\mathbf{p}) = \sum_{i=1}^{i \leq n_d} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}_i, \mathbf{y}_i). \quad (14)$$

Finally, descriptor similarity is computed using Hamming distance between the two bit strings.

ORB: Also known as Oriented FAST and rotated BRIEF, ORB descriptors (Ruble et al., 2011) are based on BRIEF and have been proposed to be robust toward noise as well as be rotationally invariant. They are primarily based on FAST key point detector (Rosten and Drummond, 2006) and extend the BRIEF descriptor (Calonder et al., 2010).

FREAK: Fast Retina Keypoint (FREAK) descriptors proposed in Alahi et al. (2012) build upon the motivation that measuring intensity difference across pixel pairs can often lead to good enough matching between image patches. They address this question of how to select the pixel pairs efficiently that is inspired by human visual system. They use the retinal sampling grid which is essentially a circle with the difference of having higher density of points near the center which drops exponentially.

4. Matching images

4.1. Histogram comparison

Histogram comparison is a popular technique to match images since most of the features extracted from images are represented as histogram values such as color histogram, texture histogram, bag-of-words, etc. Bin-wise comparison and cross-bin comparison are two primary modes of comparing histograms. In bin-wise comparison two histograms are compared bin by bin, leading to a faster way for computing (dis)similarity between two histograms. One major drawback of this method is its inability to account for similarity across bins. Hence even with small distortions such as lighting variations where histogram values are perturbed a little, bin-wise comparison will still ignore the correlation between bins and generate a higher matching cost. On the other hand, cross-bin comparison takes bin similarity into account and hence is most robust to small variations in histograms. However, cross-bin comparison methods have a higher computational cost. Some of the commonly used bin-wise and cross-bin comparison methods are in Table 1.

4.2. Geometric matching

State-of-the-art CBIR systems use Bag-of-Words (BoW) model along with local features. One main drawback of BoW models is that they do not model spatial layout.

Table 1
Popular schemes to compare two histograms H_1, H_2 that are normalized to have unit sum

	Method	Expression	Comments
Bin-wise comparison	Histogram intersection	$d_{\cap}(H_1, H_2) = \sum_{i=1}^M \min(H_1^i, H_2^i)$	Calculates the intersection of two histograms and ignores any features exclusive to a particular histogram
	Minkowski distance	$d_p(H_1, H_2) = \left(\sum_{i=1}^M (H_1^i - H_2^i)^p \right)^{\frac{1}{p}}$	Reduces to Manhattan distance for $p = 1$, Euclidean distance for $p = 2$, and maximum distance for $p = \infty$
	χ^2 -Distance	$d_{\chi^2}(H_1, H_2) = \frac{1}{2} \left[\sum_{i=1}^M \frac{(H_1^i - H_2^i)^2}{(H_1^i + H_2^i)} \right]$	Similar to weighted Euclidean distance
	Kullback–Leibler divergence	$d_{KL}(H_1, H_2) = \sum_{i=1}^M H_1^i \log \frac{H_1^i}{H_2^i}$	This is not a metric since it is asymmetric
	Jensen–Shannon divergence	$d_{JS}(H_1, H_2) = \begin{cases} \sum_{i=1}^M H_1^i \log \frac{2H_1^i}{H_1^i + H_2^i} \\ + H_2^i \log \frac{2H_2^i}{H_1^i + H_2^i} \end{cases}$	Symmetric extension of KL divergence
	Hellinger distance	$d_H(H_1, H_2) = \sqrt{1 - \sum_{i=1}^M \sqrt{H_1^i \cdot H_2^i}}$	Hellinger distance is a metric and is related to Bhattacharya coefficient by $d_H(H_1, H_2) = \sqrt{1 - BC(H_1, H_2)}$
Cross-Bin comparison	Quadratic form distance (Hafner et al., 1995)	$d_A(H_1, H_2) = \sqrt{(H_1 - H_2)^T A (H_1 - H_2)}$ where A is the bin-similarity matrix	This distance metric generalizes to Mahalanobis distance when bin-similarity matrix A is the inverse of the covariance matrix (Pele and Werman, PeleECCV2010). If A is positive-definitive, then d_A is the $L2$ norm between linear transforms of histograms H_1 and H_2 . It is also a semi-metric when A is positive semi-definite
	Earth movers distance (EMD)	$d_{EMD}(H_1, H_2) = \frac{\min_{F_{ij} \geq 0} \sum_{i,j} F_{ij} D_{ij}}{\sum_{i,j} F_{ij}}$ s.t. $\sum_j F_{ij} \leq P_i,$ $\sum_i F_{ij} \leq Q_j,$ $\sum_{i,j} F_{ij} = \min \left(\sum_i P_i, \sum_j Q_j \right)$	F_{ij} represents the amount transferred from the i th bin of H_1 to the j th bin of H_2 . D_{ij} is referred to as the “ground distance” between the bins

Modified versions BoW are used in practice so that spatial information is encoded. In Lazebnik et al. (2006), the image is recursively partitioned into uniform blocks and features are extracted from each block. These features are then concatenated to get the final feature vector. This encodes spatial relations only weakly. If speed is not a big constraint, a geometry verification step may be applied to the retrieved results as in Philbin et al. (2007), where the top retrieved images are sorted based on how well the spatial relations of interest points match with those of the query image. Another approach is to use geometry-preserving visual phrases as introduced in Zhang et al. (2011). A geometric visual phrase (GVP) of length k is defined to be a set of k visual words in a specific spatial configuration. Just like BoW uses histogram of visual words, this approach uses histogram of geometry-preserving phrases. The key lies in efficient construction and matching of these phrases. Similarity score between the query image and an image from database is determined by construction of an offset space. Matching pairs of interest points from the image pair are found. The location offset of these matching pairs are recorded in the offset space. The number of votes in an offset bin indicates the length of co-occurring geometric visual phrase. For example, in Fig. 9, offset $(-1, -1)$ contains matching points p_1, p_6 . Thus, this cell corresponds to a visual phrase of length 2. In Fig. 9, two or more visual words occur in three cells $(-1, -1), (3, 2), (0, 1)$ with points $\{p_1, p_6\}, \{p_4, p_5\}, \{p_1, p_2, p_3\}$, respectively. The number of co-occurring GVP of length 2 is then $1 + 1 + C_2^3 = 5$, where the contributions are from the three respective cells. Let $V^k(I)$ be the vector representation of the histogram of GVP of length k , with i th element the frequency of occurrence of i th GVP. It can be shown that the inner product of $V^k(I_q)$ and $V^k(I_i)$ is the total number of co-occurring GVP in this image pair. In our example, this is five. Note that $\|V^k(I_q)\|_2$ can be calculated by the frequency of co-occurring GVP with itself. In order to get cosine similarity between $V^k(I_q)$ and $V^k(I_i)$ we

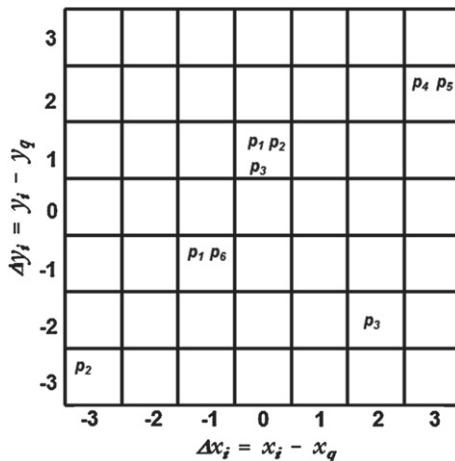


Fig. 9. Illustration of use of offset space to compute similarity between query image I_q and i th image I_i from database, based on geometric visual phrase (GVP). Note that the same visual word can appear in an image multiple times. In this example, point p_1 is matched twice in the image pair at different offsets. See Section 4.2 for more details.

need to divide inner product of $V^k(I_q)$ and $V^k(I_i)$ by the product of $\|V^k(I_q)\|_2$ and $\|V^k(I_i)\|_2$. This scheme along with inverted file look-up has the same computational complexity as BoW.

4.3. Query expansion

Query expansion—QE (popular in text retrieval community) is a technique used to improve search precision. The basic idea is to use results from initial query to reformulate the query and perform a second-pass of search to obtain higher precision results. Given its natural appeal to user interactive systems and applicability to large-scale search, recently few attempts have been made from vision perspective as well to adapt this idea to large-scale image search. We summarize some of these works below:

Richer query model: In the works of Chum et al. (2007), bag-of-visual-words is used to retrieve a set of image regions matching the query object. Multiple retrieved regions are then combined with the query to form a “richer latent model of the object of interest.” Finally, a second-pass of search is performed using the expanded query. A number of techniques such as “Transitive Closure Expansion,” “Average Query Expansion,” “Recursive Average Query Expansion,” and “Multiple Image Resolution Expansion” are investigated for this purpose. Chum et al. (2011) builds upon this work and specifically studies the question of tf-idf failures caused by the presence of sets of correlated features also referred to as “confusers.” They also propose a robust spatial verification and re-ranking step and integrate learnt spatial context for improving the search results.

Discriminative query expansion: Arandjelovic and Zisserman (2012) describe a novel discriminative query expansion technique where positive data in form of BoW vectors obtained through average query expansion and negative data represented by BoW vectors with low tf-idf scores are used to train a linear SVM. The learnt weights w are used to re-rank input images with BoW vector x by sorting on the values of $w^T x$. To provide a real-time performance, they also propose performing this “scalar product” operation with inverted indices.

5. Practical considerations: memory footprint and speed

Handling large data becomes a problem when storage and computational resources of the device are limited. This is especially true for mobile devices. To give some perspective, one million uncompressed SIFT features require about 1 TB of storage. This has spurred research interest in compacting high-dimensional features for faster retrieval, without losing much of discriminating power.

5.1. Does algorithm performance depend on size of data?

Before we delve into details of compaction of high dimensional features, let us see how performance of algorithms depend on size of data. Authors of Deng et al. (2010) show that a state-of-the-art classifier, linear SVM, outperforms simple k-Nearest Neighbor (kNN) classifier for small data sets. As size of data set increases, kNN outperforms linear SVM. In their experiment, the authors also increased the

number of categories from 7404 to 10,184. This behavior of why simple algorithm like kNN works well for large data is not fully understood. Similar observations were found in [Banko and Brill \(2001\)](#), [Torralba et al. \(2008\)](#), and [Brants et al. \(2007\)](#). As we mentioned earlier, with advancement in portable digital devices and social networking, an exponentially increasing amount of varying visual data is being created every day along with useful meta data. How can we handle this deluge of data?

5.2. Nearest neighbor

We will focus our discussion on kNN for large-scale visual search. Recall that kNN classifies samples based on proximity of samples in feature space. One major drawback of kNN is that feature vectors of all training samples are required during classification/retrieval. This means that feature vectors of *all* training samples must be stored during classification or retrieval time. As the size of training set increases, matching time increases as well. Thus, large memory footprint and reduced speed are inherent to kNN as data size increases.

From now on, let us assume k in kNN to be 1. In other words, we are interested in the nearest neighbor (NN) problem. Given a metric space X with distance function $d(\cdot, \cdot)$, the nearest neighbor problem is to find the closest point in a set of n points $P = \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n \in X$, to the query $\mathbf{q} \in X$. Let us assume X is m -dimensional real space (i.e.,) $X = \Re^m$. For large m , simple linear search is as good as any other algorithm. But, linear search is costly for large data sets in high-dimensional space.

Several algorithms have been proposed for efficient NN. One of the most popular approaches is kd-tree ([Friedman et al., 1977](#)) which has low memory footprint. kd-Tree is a binary search tree used to organize points in k -dimensional (kd) space. Each level of a kd-tree partitions the data along a dimension such that the data is divided into two halves with same number of points. A hyper plane is selected at each level in a round robin fashion. An example is shown in Fig. 10. A kd-tree can

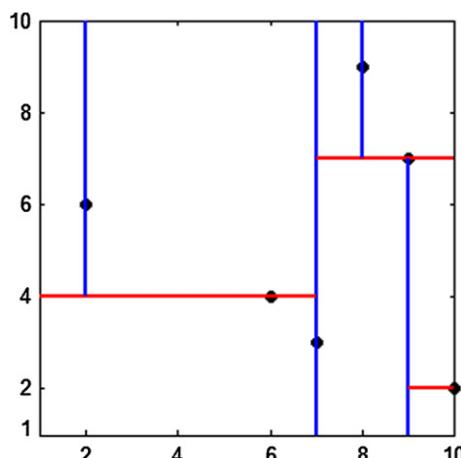


Fig. 10. Illustration of kd-tree. Axes are split alternatively into two parts with equal number of points. Axis with the largest variance is selected for the first split.

be built in $O(n \log n)$ time. Once the tree is constructed, it can be used to find the nearest neighbor of query quickly, without evaluating distance to all points in P .

5.3. Approximate nearest neighbor

kd-Tree is efficient when m is small. However, as m increases, the performance of kd-tree degrades quickly. This generated increased interest in approximate nearest neighbor approach, where the goal is not to get the exact nearest neighbor, but to get a result which is guaranteed to be close to the exact nearest neighbor.

The most popular formulation is ϵ -approximate nearest neighbor search, where given a query \mathbf{q} , the goal is to find a point $\mathbf{p} \in P$ such that

$$d(\mathbf{p}, \mathbf{q}) \leqslant (1 + \epsilon)d(\mathbf{p}', \mathbf{q}), \quad \forall \mathbf{p}' \in P \quad (15)$$

for some small $\epsilon > 0$. In the rest of the discussion, we will refer to ϵ -approximate nearest neighbor as approximate nearest neighbor (ANN) problem. One variation of kd-tree for ANN is randomized kd-trees (Silpa-Anan and Hartley, 2008), where multiple kd-trees are constructed by randomly picking among the top few (say, 5) dimensions picked based on data variance. These trees are not fully constructed, but the points in each partition are sorted based on distance from the corresponding splitting plane. When a query is supplied, matching leaf nodes are obtained by application of tree structure, and only those points in matching leaf nodes are compared. The user can specify precision of search to decide how many points from leaf nodes will be compared.

5.4. Compact binary codes

Aside from tree-based search, which does not scale well for large feature dimensions, several attempts have been made to reduce memory footprint by using compact binary codes. This idea of using compact binary codes for large-scale search was first emphasized and introduced to the computer vision community by Torralba et al. (2008). Dimensionality reduction (PCA (Mikolajczyk and Schmid, 2005), ICA (Narozny et al., 2008)), quantization (Nistér and Stewénius, 2006; Jegou et al., 2010), and hashing (LSH (Gionis et al., 1999; Wang et al., 2012)) are some of the popular families of techniques to generate compact binary codes.

Dimensionality reduction approaches are primarily unsupervised, aiming to preserve metric similarity rather than semantic similarity, because of which they do not always produce binary codes of optimal length. Dimensionality reduction approaches are also computationally expensive, thus not ideal for large-scale data.

Quantization approach in Jegou et al. (2010) is well suited for Bag-of-Words (BoW) descriptors. Coarse quantization is first done by Voronoi partitioning using k-means. Each Voronoi cell is then finely partitioned by orthogonal hyper-planes. Distance between descriptors is evaluated when descriptors lie in the same Voronoi cell. Hamming distance between the fine quantization using hyper-planes is used to approximate Euclidean distance. However, the coarse quantization steps involve k-means which is not efficient for large data sets.

Locality Sensitive Hashing (Gionis et al., 1999), LSH in short, is an early method for hashing that can find approximate nearest neighbor in constant time without

embeddings. Here, hash functions are chosen such that collision probability is small when distance between a pair of points is small and vice versa. Two points are said to collide (or fall in the same bucket) when they have the same hash code. This key property of hash function is called locality sensitivity as in:

$$\Pr_{h \in \mathcal{H}}[h(\mathbf{x}_i) = h(\mathbf{x}_j)] = \text{sim}(\mathbf{x}_i, \mathbf{x}_j), \quad (16)$$

where $h(\cdot)$ is the hash function drawn from a family \mathcal{H} of hash functions and $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$ is the similarity function of interest. The input query is mapped to a binary vector by a random set of hashing functions. All points in the same bucket as the input query are returned as similar. Additionally, these points may be sorted based on true distance to the query. This additional step can be computationally expensive if the number of points in the selected bucket is large.

Hash functions in LSH are generated from a family of p -stable distributions (Zolotarev, 1986), where $p \in [0, 2]$. Examples of p -stable distributions are Cauchy ($p = 1$) and Gaussian ($p = 2$). Given an m -dimensional vector $\mathbf{x} \in l_p^m$, its norm $\|\mathbf{x}\|_p$ can be estimated (Indyk, 2006) by a small collection of dot products $\mathbf{w}^T \mathbf{x} \in \Re$, where the m components of \mathbf{w} are generated at random from a p -stable distribution. By p -stability, given vectors $\mathbf{x}_1, \mathbf{x}_2 \in l_p^m$, the distance between their projections $\|\mathbf{w}^T \mathbf{x}_1 - \mathbf{w}^T \mathbf{x}_2\|_p$ is distributed as $\|\mathbf{x}_1 - \mathbf{x}_2\|_p D$, where D is a p -stable distribution. If \Re is quantized into regular bins of width r , the hash functions can be defined as

$$h_{\mathbf{w}, b}(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x} + b), \quad (17)$$

where b is chosen uniformly from the range $[0, r]$.

Geometrically, this means that the feature space is partitioned by randomly generated hyper-planes $\mathbf{w}^T \mathbf{x} + b = 0$, where the m components of \mathbf{w} are generated at random from a p -stable distribution (say, Gaussian) and b is chosen from uniform distribution in $[0, r]$. Bits are assigned per hyper-plane, based on which side the point lies. Length of code word is equal to the number of hyper-planes used to partition the space.

LSH guarantees asymptotic preservation of original distance metric as the length of code word increases, at the cost of reduced recall. Collision probability of codes reduces as code length increases, thus reducing recall rate. Typically, code words are long for reasonable precision. This is not favorable for large data sets where memory constraints are severe. It is important to note that hashing functions in LSH are drawn independently from data. Thus, information carried by each bit is not optimized to the data. Recent methods use data-dependent techniques to learn hash functions to minimize Hamming distance on similar data pairs and maximize on dissimilar pairs. Several flavors of such supervised methods exist. Supervised approaches produce shorter binary codes for the same performance than unsupervised approaches. Unlike unsupervised approaches that handle metric similarity, supervised approaches try to optimize semantic similarity. The details of supervised approaches are beyond the scope of this chapter. However, Table 2 has a nice comparison of various hashing techniques.

Table 2

Comparison of binary encoding methods. This table is taken from (Wang et al., 2012). LSH = Locality Sensitive Hashing, SIKH = Shift Invariant Kernel-based Hashing, SH = Spectral Hashing, RBMs = Restricted Boltzmann Machines, BRE = Binary Reconstructive Embedding, LAMP = Label-regularized Max-margin Partition, LSH-LM = LSH for Learned Metrics, SSH = Semi-Supervised Hashing

Method	Hash Function	Projection	Hamming Distance	Learning Paradigm
LSH (Gionis et al., 1999)	$\text{sgn}(\mathbf{w}^T \mathbf{x} + b)$	data-independent	non-weighted	unsupervised
SIKH (Raginsky and Lazebnik, 2009)	$\text{sgn}(\cos(\mathbf{w}^T \mathbf{x} + b) + t)$	data-independent	non-weighted	unsupervised
SH (Weiss et al., 2008)	$\text{sgn}(\cos(k\mathbf{w}^T \mathbf{x}))$	data-dependent	non-weighted	unsupervised
RBM (Hinton and Salakhutdinov, 2006)	—	—	non-weighted	supervised
BRE (Kulis and Darrell, 2009)	$\text{sgn}(\mathbf{w}^T \mathbf{k}_x)$	data-dependent	non-weighted	supervised
LAMP (Mu et al., 2010)	$\text{sgn}(\mathbf{w}^T \mathbf{x} + b)$	data-dependent	non-weighted	supervised
LSH-LM (Kulis et al., 2009)	$\text{sgn}(\mathbf{w}^T \mathbf{G}\mathbf{x})$	data-dependent	non-weighted	supervised
SSH (Wang et al., 2012)	$\text{sgn}(\mathbf{w}^T \mathbf{x} + b)$	data-dependent	non-weighted	semi-supervised

6. Benchmark data sets

Understanding the nature and scope of image data plays a key role in the complexity of image search system design (Datta et al., 2008). To achieve this goal, a data set with a large number of categories spanning a wide range of concepts and containing many images is required. Over the past decade, significant work has been put for data collection. Up to date, a wide range of data sets have been collected through various ways. This includes personal collection, domain-specific collection, enterprise collection, archives, and web, especially using current search engines such as Bing, Google, and Yahoo! images. The data are collected as a recognition process, i.e., images are obtained by using text query terms via online search engine or giant social photography sites such as Flickr. By leveraging existing online image resources, it makes it possible to collect much larger data sets with annotations (Yanai and Barnard, 2005; Li and Li, 2010; Schroff et al., 2011). Also, with the advent of Amazon Mechanical Turk, human annotation for large-scale data set becomes practical. Collected in this manner, many data sets are suitable for various purposes, such as object recognition, image classification, scene understanding, location detection, as well as image retrieval. From small to large, coarse to fine, unstructured to semi-structured, nowadays, research in computer vision has reached a new level of scale and there is more demand for large-scale applications.

Tables 3 and 4 summarize various data sets in chronological order. This is by no means the complete list, but captures different domains or types of retrieval and classification problems. The original goal of these data sets is to focus on a specific

Table 3
Benchmark data sets: 2005–2009

Name	Year	Content
Palmprint (Sun et al., 2005)	2005	502 palmprint images from 312 subjects
UIUC Textures (Lazebnik et al., 2005)	2005	25 classes, 40 images per class. Contains materials imaged under significant viewpoint variations and contains fabrics which display folds and have non-rigid surface deformations
Animals on the Web (Berg and Forsyth, 2006)	2006	14,051 images of 10 different animals from 9972 web pages using Google text search. In addition, contains 12,866 images of monkeys out of which only 2569 are actual monkeys
UKBench (Nistér and Stewénius, 2006)	2006	10,200 images of 2550 objects where each object has four images taken from different viewpoints and illuminations
Caltech 256 (Griffin et al., DatasetCaltech2560)	2007	30,607 images from 256 categories, minimum 80 images per category
Labeled Faces in the Wild (Huang et al., 2007)	2007	13,223 images, 5749 people, 1680 images with two or more images
Oxford buildings (Philbin et al., 2007)	2007	5062 images from Flickr containing particular Oxford landmarks. Ground truth for 11 landmarks with 5 queries per landmark. Contains outline of objects
ETHZ Extended Shape (Schindler and Suter, 2008)	2008	7 shape classes with 50 images from each class. The images may contain multiple instances of a category and have a large amount of background clutter
Flowers (Nilsback and Zisserman, 2008)	2008	8189 images containing 102 species of flowers with 40–258 images each, from various websites, with varying pose, light, scale. A subset of images have ground truth for segmentation
Im2GPS (Hays and Efros, 2008)	2008	Over 6M GPS-tagged images
INRIA Holidays (Jégou et al., 2008)	2008	1491 images of personal holiday photos which are grouped into 500 sets with one query per set
LabelMe (Russell et al., 2008)	2008	Over 187k images of which over 62k are annotated. There are more than 659k labeled objects. Multiple objects in scene are annotated by a polygonal bounding box
Paris (Philbin et al., 2008)	2008	6412 images from Flickr containing particular Paris landmarks. Same format as Oxford landmarks data set
Tiny Images (Torralba et al., 2008)	2008	79,302,017 color images of size 32×32 , loosely labeled with one of 75062 nouns in English as in WordNet (Miller, 1995)
Animals with attributes (Lampert et al., 2009)	2009	30,475 images of 50 animals classes with 85 numeric attributes for each class
Butterflies (Wang et al., 2009)	2009	832 images of 10 different species of butterflies, with 55–100 images per category
ImageNet (Deng et al., 2009)	2009	14,197,122 images grouped hierarchically based on synonym sets from WordNet (Miller, 1995). Over 1M images have bounding box annotations

problem and come up with strategies to solve that problem. Progressively, subsequent data sets were created to increase the complexity of the problem posed by previous data sets. Authors of Torralba and Efros (2011) emphasize that due to overuse of these data sets, algorithms could overfit to specific data sets and not generalize. They built classifiers to recognize various popular data sets and analyze the bias

Table 4
Benchmark data sets: 2010–2012

Name	Year	Content
CASIA-FingerprintV5 (Tan and Sun, 2010)	2010	20,000 fingerprint images of 500 subjects
CASIA-IrisV4 (Tan and Sun, 2010)	2010	54,601 iris images from more than 1800 genuine subjects and 1000 virtual subjects
European Cities 1M (Avrithis et al., 2010)	2010	909,940 geo-tagged images from 22 European cities
MIRFLICKR-1M (Mark et al., 2010)	2010	1 million high quality Flickr images with topics, subtopics, keywords, and EXIF metadata
SUN (Xiao et al., 2010)	2010	130519 images from 899 scene categories
Web Queries (Krapac et al., 2010)	2010	71,478 images and meta-data retrieved by 353 web queries
CUB-200–2011 (Wah et al., 2011)	2011	11,788 images of birds from 200 categories. Each image contains annotations of 15 part locations, 312 binary attributes, and a bounding box
RGB-D (Lai et al., 2011)	2011	250K RGB-D images based on RGB and depth video sequences of 300 common everyday objects from multiple view angles
San Francisco Landmarks (Chen et al., 2011)	2011	1.7M perspective images created from 150K panoramic images of landmarks in San Francisco. Contains geo-tags, calibration data, and query images from mobile phone
Sculptures 6k (Arandjelović and Zisserman, 2011)	2011	6340 images of sculptures by Henry Moore and Auguste Rodin displayed at different locations, downloaded from Flickr. Ground truth query for 10 different Henry Moore sculptures with 7 images each
Stanford Dogs (Khosla et al., 2011)	2011	20,580 images of dogs from 120 breeds. Contains bounding boxes
Oxford-IIIT-Pet dataset (Parkhi et al., 2012)	2012	Images of cats and dogs from 12 cat breeds and 25 dog breeds, with about 200 images per breed. Contains tight bounding box and trimap
PASCAL VOC (Everingham et al., 2012)	2012	First introduced in 2005 and evolved periodically. 11,530 images from 20 object categories including people, animals, vehicles, indoor objects, containing 27450 ROI annotations. 9993 images have segmentation mask
Sketch (Eitz et al., 2012)	2012	20,000 sketch images equally distributed over 250 objects
TRECVID (Over et al., 2012)	2012	About 200 h worth of 8000 internet archive videos, with duration between 10 s and 3.5 min each. Most videos have title, keywords, and description

introduced by these data sets. Nevertheless, considerable progress has been made through these data sets as they could be used to train image retrieval/classification systems and to compare performance of various algorithms.

7. Closing remarks

In this chapter, we summarized different use-cases for vision-based retrieval, information extraction from images, matching extracted information, memory and

speed optimization, and computer vision data sets. There are other important components that we did not cover but are essential for large-scale visual search. Some of these are highlighted below:

Processing power: Processing large data requires hardware processing power. Many computer vision algorithms deal with processing large blocks of data independently. These operations could be executed in parallel on the same device or in a distributed manner. Graphical Processing Units (GPUs) provide a data-parallel arithmetic architecture that allows similar set of calculations to be performed on a block of data. Use of GPUs for large-scale problems has been reported in Raina et al. (2009) and Nagesh et al. (2010). For distributed computing on clusters of computers, the MapReduce paradigm can be used as in White et al. (2010). A combination of using GPU clusters with MapReduce paradigm can also be used as in Stuart and Owens (2011).

One-shot learning: We emphasized the importance of large data to take into account the diversities of form and appearance of objects in the scene, as well as sufficiency of simple algorithms. However, humans can learn visual concepts with just few examples, in some cases even with a single positive example, and make meaningful generalizations. This is addressed by one-shot learning and transfer learning as in Li et al. (2006) and Salakhutdinov et al. (2011). The idea behind this is that knowledge about a category once acquired can be used to learn new categories, with only a few examples. Going back to Fig. 1, humans can interpret the objects shown to be chairs even though they might not have seen instances of these before.

Domain adaptation: Visual retrieval systems must be able to retrieve images of various renditions of matching objects. Even for a single object in a single pose, the composition of the image and camera specification can vary significantly (Fig. 3). We presented in Section 3 how information can be extracted from images. By design, few of these approaches handle variations in pose and lighting conditions. There are other ways how these variations can be handled. Metric learning and domain adaptation are examples (Kulis et al., 2011; Gong et al., 2012).

Relevance feedback: When the input query is just a single image, there is ambiguity in what aspect of the query image needs to be matched. For example, should similarity be in terms of color, texture, shape, functionality, pose, etc. or a combination of these? This missing information (either explicit or implicit) is referred to as semantic gap. Semantic gap can be reduced by requesting for more information during query time or after retrieval is complete and results are presented. In either case, information extracted from the data are weighted based on relevance and similarity metric may be learned accordingly. The goal of relevance feedback mechanism is to reduce the semantic gap by getting feedback from user to refine the retrieved result, so that the quality of retrieved results is maximized with minimal interaction between the user and the system. User feedback is in the form of selection of relevant and/or irrelevant instances from the retrieved results. One of the biggest challenges is to learn intent of query through limited number of positive examples. Examples of relevance feedback mechanisms can be found in Wang et al. (2011) and Tronci et al. (2012).

Multimodality: Large data, especially from the Internet, do not consist of just visual content such as images and videos. They also contain other useful information such as text, image meta-data (EXIF), GPS location, user information, or audio. It is

important to harness this diversity of information so that relevant documents can be achieved. Blei and Jordan (2003) and Li et al. (2009) show how to integrate visual information with text.

We hope that the material presented in this chapter is broad and detailed enough to appreciate the complexity of CBIR and motivate different areas where statistical machine learning tools could be applied for advancement.

References

- Alahi, A., Ortiz, R., Vandergheynst, P., 2012. Freak: fast retina keypoint. In: Conference on Computer Vision and Pattern Recognition, pp. 510–517.
- Arandjelović, R., Zisserman, A., 2011. Smooth object retrieval using a bag of boundaries. In: IEEE International Conference on Computer Vision.
- Arandjelovic, R., Zisserman, A., 2012. Three things everyone should know to improve object retrieval. In: Conference on Computer Vision and Pattern Recognition, pp. 2911–2918.
- Avrithis, Y., Kalantidis, Y., Tolias, G., Spyrou, E., 2010. Retrieving landmark and non-landmark images from community photo collections. In: Proceedings of ACM Multimedia, Firenze, Italy.
- Banko, M., Brill, E., 2001. Scaling to very very large corpora for natural language disambiguation. In: 39th Annual Meeting on Association for Computational Linguistics, pp. 26–33.
- Bay, H., Tuytelaars, T., Gool, L.J.V., 2006. Surf: speeded up robust features. In: European Conference on Computer Vision, vol. 1, pp. 404–417.
- Bellman, R., 1957. Dynamic Programming, first ed. Princeton University Press, Princeton, NJ, USA.
- Belongie, S., Malik, J., Puzicha, J., 2000. Shape context: a new descriptor for shape matching and object recognition. In: The Conference on Neural Information Processing Systems, pp. 831–837.
- Berg, T.L., Forsyth, D.A., 2006. Animals on the web. In: Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1463–1470.
- Blei, D.M., Jordan, M.I., 2003. Modeling annotated data. In: Special Inspector General for Iraq Reconstruction, pp. 127–134.
- Brants, T., Popat, A.C., Xu, P., Och, F.J., Dean, J., 2007. Large language models in machine translation. In: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language, 2007, pp. 858–867.
- Calonder, M., Lepetit, V., Strecha, C., Fua, P., 2010. Brief: binary robust independent elementary features. In: European Conference on Computer Vision, vol. 4, pp. 778–792.
- Caron, Y., Makris, P., Vincent, N., 2007. Use of power law models in detecting region of interest. Pattern Recogn. 40, 2521–2529.
- Chen, D.M., Baatz, G., Köser, K., Tsai, S.S., Vedantham, R., Pylvänen, T., Roimela, K., Chen, X., Bach, J., Pollefeyt, M., Girod, B., Grzeszczuk, R., 2011. City-scale landmark identification on mobile devices. In: Conference on Computer Vision and Pattern Recognition, pp. 737–744.
- Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A., 2007. Total recall: automatic query expansion with a generative feature model for object retrieval. In: International Conference on Computer Vision, pp. 1–8.
- Chum, O., Mikulík, A., Perdoch, M., Matas, J., 2011. Total recall ii: query expansion revisited. In: Conference on Computer Vision and Pattern Recognition, pp. 889–896.
- Cinque, L., Levialdi, S., Pellicanò, A., Olsen, K.A., 1999. Color-based image retrieval using spatial-chromatic histograms. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, ICMCS 1999, vol. 2. IEEE Computer Society, Washington, DC, USA, pp. 969–973.
- Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, pp. 886–893.
- Datta, R., Joshi, D., Li, J., Wang, J.Z., 2008. Image retrieval: ideas, influences, and trends of the new age. ACM Comput. Surv. (CSUR) 40 (2).
- Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. ImageNet: a large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition09.

- Deng, J., Berg, A.C., Li, K., Li, F.F., 2010. What does classifying more than 10,000 image categories tell us? In: European Conference on Computer Vision, vol. 5, pp. 71–84.
- Eitz, M., Hays, J., Alexa, M., 2012. How do humans sketch objects? ACM Trans. Graph. (Proc. SIGGRAPH) 31, 44:1–44:10.
- Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2012. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <<http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>>.
- Friedman, J.H., Bentley, J.L., Finkel, R.A., 1977. An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Softw. 3, 209–226.
- Funt, B.V., Finlayson, G.D., 1995. Color constant color indexing. IEEE Trans. Pattern Anal. Mach. Intell. 17, 522–529.
- Georghiades, A., Belhumeur, P., Kriegman, D., 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell. 23, 643–660.
- Gionis, A., Indyk, P., Motwani, R., 1999. Similarity search in high dimensions via hashing. In: International Conference on Very Large Data Bases, pp. 518–529.
- Gong, B., Shi, Y., Sha, F., Grauman, K., 2012. Geodesic flow kernel for unsupervised domain adaptation. In: Conference on Computer Vision and Pattern Recognition, pp. 2066–2073.
- Graczyk, C., 1995. Vision texture. <<http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html>>.
- Griffin, G., Holub, A., Perona, P., 2007. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology.
- Hafner, J.L., Sawhney, H.S., Equitz, W., Flickner, M., Niblack, W., 1995. Efficient color histogram indexing for quadratic form distance functions. IEEE Trans. Pattern Anal. Mach. Intell. 17, 729–736.
- Halevy, A.Y., Norvig, P., Pereira, F., 2009. The unreasonable effectiveness of data. IEEE Intell. Syst. 24, 8–12.
- Hays, J., Efros, A.A., 2008. im2gps: estimating geographic information from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- Hinton, G.E., Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. Science 313, 504–507.
- Hu, M.K., 1962. Visual pattern recognition by moment invariants. IRE Trans. Inform. Theory 179–187.
- Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E., 2007. Labeled faces in the wild: a database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst.
- Indyk, P., 2006. Stable distributions, pseudorandom generators, embeddings, and data stream computation. J. ACM (JACM) 53 (3), 307–323.
- Jégou, H., Douze, M., Schmid, C., 2008. Hamming embedding and weak geometric consistency for large-scale image search. In: David Forsyth, Philip Torr, A.Z. (Eds.), European Conference on Computer Vision. LNCS, vol. I. Springer, pp. 304–317.
- Jégou, H., Douze, M., Schmid, C., 2010. Improving bag-of-features for large-scale image search. Int. J. Comput. Vis. 87, 316–336.
- Khosla, A., Jayadevaprakash, N., Yao, B., Fei-Fei, L., 2011. Novel dataset for fine-grained image categorization. In: First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition, Colorado Springs, CO.
- Kokkinos, I., Maragos, P., Yuille, A.L., 2006. Bottom-up and top-down object detection using primal sketch features and graphical models. In: Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1893–1900.
- Krapac, J., Allan, M., Verbeek, J.J., Jurie, F., 2010. Improving web image search results using query-relative classifiers. In: Conference on Computer Vision and Pattern Recognition, pp. 1094–1101.
- Kulis, B., Darrell, T., 2009. Learning to hash with binary reconstructive embeddings. In: Advances in Neural Information Processing Systems, vol. 22, pp. 1042–1050.
- Kulis, B., Jain, P., Grauman, K., 2009. Fast similarity search for learned metrics. IEEE Trans. Pattern Anal. Mach. Intell. 2143–2157.
- Kulis, B., Saenko, K., Darrell, T., 2011. What you saw is not what you get: domain adaptation using asymmetric kernel transforms. In: Conference on Computer Vision and Pattern Recognition. IEEE, pp. 1785–1792.

- Lai, K., Bo, L., Ren, X., Fox, D., 2011. A large-scale hierarchical multi-view rgb-d object dataset. In: ICRA, pp. 1817–1824.
- Lampert, C.H., Nickisch, H., Harmeling, S., 2009. Learning to detect unseen object classes by between-class attribute transfer. In: Conference on Computer Vision and Pattern Recognition, pp. 951–958.
- Lazebnik, S., Schmid, C., Ponce, J., 2005. A sparse texture representation using local affine regions. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1265–1278.
- Lazebnik, S., Schmid, C., Ponce, J., 2006. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178.
- Levin, A., Weiss, Y., 2009. Learning to combine bottom-up and top-down segmentation. *Int. J. Comput. Vis.* 81, 105–118.
- Li, L.J., Li, F.F., 2010. Optimol: automatic online picture collection via incremental model learning. *Int. J. Comput. Vis.* 88, 147–168.
- Li, F.F., Fergus, R., Perona, P., 2006. One-shot learning of object categories, *IEEE Trans. Pattern Anal. Mach. Intell.* 28, 594–611.
- Li, L.J., Socher, R., Li, F.F., 2009. Towards total scene understanding: classification, annotation and segmentation in an automatic framework. In: Conference on Computer Vision and Pattern Recognition, pp. 2036–2043.
- Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* 60, 91–110.
- Maenpaa, T., 2003. The Local Binary Pattern Approach to Texture Analysis—Extensions and Applications. Academic Dissertation, Infotech Oulu and Department of Electrical and Information Engineering, University of Oulu.
- Manjunath, B.S., Ma, W.Y., 1996. Texture features for browsing and retrieval of image data. *IEEE Trans. Pattern Anal. Mach. Intell.* 18, 837–842.
- Mark J. Huiskes, B.T., Lew, M.S., 2010. New trends and ideas in visual concept detection: the MIR Flickr retrieval evaluation initiative. In: Proceedings of the ACM International Conference on Multimedia Information Retrieval 2010, MIR 2010, ACM, New York, NY, USA, pp. 527–536.
- Mikolajczyk, K., Schmid, C., 2005. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1615–1630.
- Miller, G.A., 1995. Wordnet: a lexical database for English. *Commun. ACM* 38, 39–41.
- Mu, Y., Shen, J., Yan, S., 2010. Weakly-supervised hashing in kernel space. In: Conference on Computer Vision and Pattern Recognition, pp. 3344–3351.
- Nagesh, P., Gowda, R., Li, B., 2010. Fast gpu implementation of large-scale dictionary and sparse representation based vision problems. In: International Conference on Acoustics, Speech, and Signal Processing, pp. 1570–1573.
- Narozny, M., Barret, M., Pham, D.T., 2008. Ica based algorithms for computing optimal 1-d linear block transforms in variable high-rate source coding. *Signal Process.* 88, 268–283.
- Newman, M.E.J., 2005. Power laws, pareto distributions and zipf's law. *Contemp. Phys.* 46 (5), 323–351.
- Nilsback, M.E., Zisserman, A., 2008. Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing.
- Nistér, D., Stewénius, H., 2006. Scalable recognition with a vocabulary tree. In: Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2161–2168.
- Nowak, L.G., Bullier, J., 1997. The timing of information transfer in the visual system. In: Extrastriate Cortex in Primates, pp. 205–241.
- Ojala, T., Pietikäinen, M., Harwood, D., 1996. A comparative study of texture measures with classification based on feature distributions. *Pattern Recogn.* 29, 51–59.
- Oliva, A., Torralba, A., 2006. Building the gist of a scene: the role of global image features in recognition. *Prog. Brain Res.* 155, 23–36.
- Over, P., Awad, G., Michel, M., Fiscus, J., Sanders, G., Shaw, B., Kraaij, W., Smeaton, A.F., QuTenot, G., 2012. Trecvid 2012—an overview of the goals, tasks, data, evaluation mechanisms and metrics. In: Proceedings of TRECVID 2012, NIST, USA.
- Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.V., 2012. Cats and dogs. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Pele, O., Werman, M., 2010. The quadratic-chi histogram distance family. In: European Conference on Computer Vision, vol. 2, pp. 749–762.

- Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2007. Object retrieval with large vocabularies and fast spatial matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
- Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2008. Lost in quantization: Improving particular object retrieval in large-scale image databases. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
- Raginsky, M., Lazebnik, S., 2009. Locality-sensitive binary codes from shift-invariant kernels. In: The Conference on Neural Information Processing Systems, pp. 1509–1517.
- Raina, R., Madhavan, A., Ng, A.Y., 2009. Large-scale deep unsupervised learning using graphics processors. In: International Conference on Machine Learning, p. 110.
- Rosten, E., Drummond, T., 2006. Machine learning for high-speed corner detection. In: Proceedings of the 9th European conference on Computer Vision ECCV 2006, vol. Part I. Springer-Verlag, Berlin, Heidelberg, pp. 430–443.
- Rublee, E., Rabaud, V., Konolige, K., Bradski, G.R., 2011. Orb: an efficient alternative to sift or surf. In: International Conference on Computer Vision, pp. 2564–2571.
- Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T., 2008. Labelme: a database and web-based tool for image annotation. *Int. J. Comput. Vis.* 77, 157–173.
- Salakhutdinov, R., Tenenbaum, J., Torralba, A., 2011. One-shot learning with a hierarchical nonparametric bayesian model. In: JMLR: Workshop on Unsupervised and Transfer, Learning, pp. 1068–1079.
- Schindler, K., Suter, D., 2008. Object detection by global contour shape. *Pattern Recognit.* 41, 3736–3748.
- Schroff, F., Criminisi, A., Zisserman, A., 2011. Harvesting image databases from the web. *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (4), 754–766.
- Silpa-Anan, C., Hartley, R., 2008. Optimised kd-trees for fast image descriptor matching, In: Conference on Computer Vision and Pattern Recognition.
- Sivic, J., Zisserman, A., Zisserman, A., 2003. Video google: a text retrieval approach to object matching in videos. In: International Conference on Computer Vision, pp. 1470–1477.
- Stuart, J.A., Owens, J.D., 2011. Multi-gpu mapreduce on gpu clusters. In: International Parallel and Distributed Processing Symposium, pp. 1068–1079.
- Sun, Z., Tan, T., Wang, Y., Li, S.Z., 2005. Ordinal palmprint representation for personal identification. In: Conference on Computer Vision and Pattern Recognition, pp. 279–284.
- Tamura, H., Mori, S., Yamawaki, T., 1978. Textural features corresponding to visual perception. *IEEE Trans. Syst. Man Cybern.* 8 (6), 460–472.
- Tan, T., Sun, Z., 2010. Chinese academy of sciences' institute of automation, CASIA-FingerprintV5. <<http://biometrics.idealtest.org/>>.
- Tan, T., Sun, Z., 2010. Chinese academy of sciences' institute of automation, CASIA-IrisV4. <<http://biometrics.idealtest.org/>>.
- Torralba, A., Efros, A.A., 2011. Unbiased look at dataset bias. In: Conference on Computer Vision and Pattern Recognition, pp. 1521–1528.
- Torralba, A., Fergus, R., Freeman, W.T., 2008. 80 million tiny images: a large database for non-parametric object and scene recognition. *IEEE PAMI* 30, 1958–1970.
- Torralba, A., Fergus, R., Weiss, Y., 2008. Small codes and large image databases for recognition. In: Conference on Computer Vision and Pattern Recognition.
- Tronci, R., Piras, L., Giacinto, G., 2012. Performance evaluation of relevance feedback for image retrieval by real-world multi-tagged image datasets. *Int. J. Multimed. Data Eng. Manag. (IJMDEM)* 3 (1), 1–16.
- Turk, M., Pentland, A., 1991. Eigenfaces for recognition. *J. Cogn. Neurosci.* 3, 71–96.
- Varma, M., Zisserman, A., 2005. A statistical approach to texture classification from single images. *Int. J. Comput. Vis.* 62, 61–81.
- Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S., 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.
- Wang, J., Markert, K., Everingham, M., 2009. Learning models for object recognition from natural language descriptions. In: Proceedings of the British Machine Vision Conference.
- Wang, X., Liu, K., Tang, X., 2011. Query-specific visual semantic spaces for web image re-ranking. In: Conference on Computer Vision and Pattern Recognition, pp. 857–864.
- Wang, J., Kumar, S., Chang, S.F., 2012. Semi-supervised hashing for large-scale search. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 2393–2406.

- Weiss, Y., Torralba, A., Fergus, R., 2008. Spectral hashing. In: The Conference on Neural Information Processing Systems, pp. 1753–1760.
- Wertheimer, M., 1923. Untersuchungen zur Lehre von der Gestalt. II. Psych. Forshung 4, 301–350.
- White, B., Yeh, T., Lin, J., Davis, L., 2010. Web-scale computer vision using mapreduce for multimedia data mining. In: International Workshop on Multimedia Data Mining.
- Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A., 2010. Sun database: large-scale scene recognition from abbey to zoo. In: Conference on Computer Vision and Pattern Recognition, pp. 3485–3492.
- Yanai, K., Barnard, K., 2005. Probabilistic web image gathering. In: Multimedia Information Retrieval, pp. 57–64.
- Zelinsky, G.J., Zhang, W., Yu, B., Chen, X., Samaras, D., 2005. The role of top-down and bottom-up processes in guiding eye movements during visual search. In: The Conference on Neural Information Processing Systems.
- Zhang, Y., Jia, Z., Chen, T., 2011. Image retrieval with geometry-preserving visual phrases. In: Conference on Computer Vision and Pattern Recognition, pp. 809–816.
- Zolotarev, V.M., 1986. One-dimensional Stable Distributions. Translations of Mathematical Monographs ed., vol. 65. American Mathematical Society.

This page is intentionally left blank

Part III: Biometric Systems

This page is intentionally left blank

Video Activity Recognition by Luminance Differential Trajectory and Aligned Projection Distance

*Haomian Zheng¹, Zhu Li², Yun Fu³, Aggelos K. Katsaggelos⁴,
and Jane You¹*

¹*Department of Computing, Hong Kong Polytechnic University, Hong Kong*

²*Multimedia Core Standards Research Samsung Research America,
Richardson, USA*

³*Department of ECE, College of Engineering and College of CIS (Affiliated),
Northeastern University, 360 Huntington Avenue Boston, MA 02115, USA*

⁴*Department of Electrical Engineering and Computer Science,
Northwestern University, Evanston, Illinois, USA*

Abstract

Video content analysis and understanding are active research topics in modern visual computing and communication. In this context, a particular challenging problem that attracts much attention is human action recognition. In this chapter, we propose a new methodology to solve the problem using geometric statistical information. Two new approaches, Differential Luminance Field Trajectory (DLFT) and Luminance Aligned Projection Distance (LAPD), are proposed. Instead of extracting the object or using interest points as a representation, we treat each video clip as a trajectory in a very high dimensionality space and extract the useful statistical geometric information for action recognition. For DLFT, we take advantage of the differential signals which preserve both the temporal and spatial information, and then classify the action by supervised learning. For the LAPD approach, we generate a trajectory for each video clip and compute a distance metric to describe the similarity for classification. Decision is made by applying a K-Nearest Neighbor classifier. Since DLFT is more sensitive in the temporal domain while the LAPD approach can handle more variance in appearance luminance field, a potential fusion of the two methods would yield more desirable properties. Experimental results demonstrate that the methods work effectively and efficiently. The performance is comparable or better and more robust than conventional methods.

Keywords: action recognition, gesture recognition, differential luminance field trajectory, luminance aligned projection distance

1. Introduction

With the development of computing and communication technologies, video content analysis is becoming one of the most popular research areas in computer vision and machine learning. Video action recognition has wide applicability in video surveillance, entertainment, sports, searching, human-computer interaction, and many other activities in daily life. Activity recognition is a subtopic in video pattern recognition, the problem can be defined as classifying a query action into several pre-defined classes. That is, given a sequence of video frames, with one or several subjects performing an activity, a system is required to automatically recognize the type of activity.

Generally, the set of actions contains a semantic meaning in our daily life, such as running, clapping, or jumping. Actions can be categorized into different spatio-temporal patterns according to these meanings. Video sequences consist of massive amount of raw data in the form of spatio-temporal pixel intensity variations. However, most of this information is not directly relevant to the task on understanding and identifying the activity in the video. Then a problem is determining which is related to such semantic meanings and contributive to the classification. Aggarwal and Cai (1999) focused on extracting the body structures and tracking across frames, and in Cedras and Shah (1995) focused on finding and utilizing the motion, which is believed to be more important for action recognition. Both techniques were concentrated on extracting the effective features in action recognition, which is a critical problem in this topic.

Various kinds of features such as luminance points (Hoey and Little, 2000) and human body detection (Yang et al., 2009) were proposed in the literature and demonstrated to have good performance on discriminative action recognition. Spatio-temporal interest points (Laptev and Kindeberg, 2003) were proved to be insensitive to scaling, rotation, and background variations are more frequently used than others. 3-D shapes (Gorelick et al., 2007) and 4-D action feature model (Yan et al., 2008) were developed to better represent the video clips.

Given the extracted features, another challenging problem is how to better utilize the features to achieve good performance on action recognition. The modeling of the actions can be generally categorized into three different types—*non-parametric*, *volumetric*, and *parametric time-series* approaches (Turaga et al., 2008). With non-parametric modeling, the features are extracted from each frame of the video sequence and then matched with a stored template. With volumetric approaches, the video is considered as a 3-D volume of pixel intensities based on which the volumetric models are constructed. With parametric time-series approaches, a model is imposed on the temporal dynamics of the motion. Particular parameters are then estimated from the training data, which is treated as a signature of a certain class of actions.

There are several challenging sub-problems in this area, which are the subjects of intense research. In general, these sub-problems can be categorized into low-level pre-processing, human appearance representation, and subspace learning. The low-level pre-processing applied on the original video clip leads to the information extraction which is used as a representation of the human body for action recognition. The subspace learning basically focused on finding a subspace

to project while maintaining the discrimination and at the same time reducing the dimensionality in the original signals. A subspace can be learned in various ways to train a system, which can recognize the query video clip automatically (Li et al., 2008; Laptev et al., 2008; Zheng et al., 2009).

Some human-computer interactive systems were also developed for this task. In such systems, a human can define some special interesting points to help recognition. For example, in Yang et al. (2009), the hands, feet, and head of a person are manually marked by the user before training and recognition. With this prior information, the performance is usually better than that of the fully unsupervised systems.

In this chapter, we propose two different methods, both of which are based on luminance spatio-temporal features, to recognize different actions. The first approach is the Differential Luminance Field Trajectory (DLFT). Luminance video frames are vectorized and projected into a high dimensional space, thus forming a trajectory which is a representation of the video. Differential values of the luminance trajectory are extracted as features for learning and recognition. By further processing the features, the differential signals are fully discriminated and classification is applied based on traditional classifiers. In the second approach, Luminance Aligned Projection Distance (LAPD), the trajectory is also used as a representation of the video sequences. Instead of traditional Euclidean or subspace distance computing, a new measurement for similarity is proposed by introducing an appropriate subspace and finding an optimal alignment. This alignment is used to match the two trajectories and get a reliable distance between them. A distance metric is constructed in this way and a KNN classifier can be applied for recognition. The effectiveness and robustness of our proposed methods are demonstrated by experiments on several benchmark human action datasets.

This work is an extension of our initial work, Li et al. (2009) and Zheng et al. (2009). Here we extend DLFT into a more flexible approach by introducing the DFT in the transformation step, which improves the performance on all the tested datasets as well as the flexibility of the algorithm. For LAPD we reformulate the problem into an optimal matching between trajectories. For both of our proposed approaches, we test them on two new datasets to further demonstrate the effectiveness and robustness. Furthermore, a joint DLFT and LAPD is applied to achieve a better performance than the individual algorithms. In the experiment, we compare our proposed method with some recent work in the literature. We also implement a classic algorithm, Dynamic Time Warping (DTW) for action recognition, as a baseline comparison with our proposed methods.

The chapter is organized into the following sections. In Section 2, we briefly review some of the popular techniques in the human action recognition literature. The formulation for our proposed methods is given in Section 3. Solutions for proposed methods are presented in Section 4, experimental results are shown in Section 5 and compared with other recent works. Finally conclusions are drawn in Section 6 and future work is included as well.

2. Related work

As an important area in computer vision, human action and activity recognition have received much attention in recent years. A comprehensive review of this research

topic has been presented in a number of survey papers, e.g., [Aggarwal and Cai \(1999\)](#) and [Cedras and Shah \(1995\)](#). In this section, we mainly focus on discussing the most critical processing in this special problem.

Pixel values can be directly obtained from an image or a video clip. So the optical flow representation, which is based on the moving pixels, has been widely used as a simple representation of the video by a lot of researchers ([Efros et al., 2003](#); [Hoey and Little, 2000](#)). In this approach, the idea is to directly use the optical flow to derive a video representation which can be used for recognition. So motion detection and analysis from video compression work have also been combined into this technique. For example, Motion Energy Image (MEI) ([Bobick and Davis, 1996](#)) and Motion History Image (MHI) ([Bobick and Davis, 2001](#)) have been proposed to describe the motion information.

In general, a class of approaches for human action recognition analysis is based on the modeling of the extracted features from the video sequences. The modeling and learning of the extracted features are the critical part of the problem, in improving the accuracy of the recognition. Some popular techniques include optical motion detection, 3-D volume representation, temporal modeling, Hidden Markov Model (HMM) training, Dynamic Time Warping (DTW), and multi-view subspace learning. We offer a brief review of these techniques in the following several paragraphs.

The appearance-based feature representation is not robust with respect to background changes such as scaling and rotation. Also, the failure on handling occlusions and cloth changing limited the application on these methods. Space-time interest points and their trajectories for action and activity analysis are quite popular in the recent literature ([Laptev et al., 2008](#); [Oikonomopoulous et al., 2005](#)). The main strength of this representation is the robustness to occlusions, since there is no need to detect or track the human body or hand. A dictionary can be constructed by a bag-of-words approach and therefore the image or video can be represented by the statistical information of words.

Temporal properties have been proved to contribute a lot toward action classification. Compared with traditional 3-D modeling, a 4-D (x, y, z, t) action feature model (AFM) was proposed for representation and recognition of action from arbitrary views ([Yan et al., 2008](#)). Temporal features are also highly emphasized in [Vail et al. \(2007\)](#) for creating intelligent robot systems. By utilizing Conditional Random Fields (CRF) and applying discrimination training, the algorithm is proved to be effective.

Researchers also applied Hidden Markov Models (HMMs) and their variants for better analysis of their temporal behavior ([Sundaresan et al., 2003](#)). The general methodology was to learn the appearance model of the human body or hand and match it explicitly to images in a target video sequence for action and gesture recognition ([Yamato et al., 1992](#)). This approach is highly dependent on the features extracted from the video. Different representations also have different models. In [Gorelick et al. \(2007\)](#), actions in video clip are treated as 3-D shapes induced by silhouettes in the space-time volume and properties of the solution for Poisson equation were utilized to extract feature, such as action dynamics, shape structure, and orientation. The method is proven to be fast and robust to partial occlusions and can be applied to low-quality videos. Similarly,

in Weinland et al. (2007) an exemplar-based HMM was proposed and this model took advantage of dependencies between three-dimensional exemplars. Furthermore, a template-based method, named the Maximum Average Correlation Height (MACH), was proposed in Rodriguez et al. (2008). By capturing the intra-class variability, the single action class is simply and carefully modeled after analyzing the response of the MACH filter.

Dynamic Time Warping is a traditional approach in speech recognition, which is used to align audio sequence of different duration. This technique was recently applied to human action recognition as well (Cherla et al., 2008; Pierobon et al., 2005). This method is proved to be robust to scaling and rotation, and has good performance encountered by only very low-dimensional features, but its application is limited to only a few action patterns.

Instead of building models for only one set of the features, there are some approaches that focused on both temporal and spatial domains. A more comprehensive understanding can be obtained during such a process. In Ke et al. (2005), a spatio-temporal volume modeling based solution is investigated and proved to be insensitive to image formation variations. In Wu and Yu (2006), a new approach, which is composed of a two-layer statistical field model, was proposed and demonstrated to be robust to occlusions. Besides robustness, the structure was also more flexible to image observations, which made the method robust to clutter as well. In Kim et al. (2007b) Canonical Correlation Analysis (CCA) is used to measure the similarity of any two image sets for robust object recognition. Correlation information is also considered to be helpful for recognition. After this, in Kim et al. (2007a), a method is also applied for hand gesture recognition by combining feature selection and the Tensor Canonical Correlation Analysis (TCCA) learning process. Tensor work has also been applied for gait recognition in Tao et al. (2007), combined with Gabor features contained in the gait sequence.

3. Problem formulation

In this section we describe the problem formulation, along with several pre-processing steps: video representation, noise reduction, and subspace learning. Challenges in the human action recognition tasks include human detection and representation, motion understanding and analysis. By solving these problems with appropriate algorithms, the signals can be prepared for learning and recognition.

Video clips are composed of frames which consist of pixel values. It is a challenging task to detect the human body in video sequences, especially with large visual variations and occlusions. Originally, researchers treated human as a single object in the frame so that the human body can be separated from the background. A number of solutions based on this idea have already been proposed in the literature. Traditional methods focused on detecting and recognizing different human actions, such as in Cherla et al. (1998) and Zelnik-Manor and Irani (2001). The main techniques involved are the so-called “object-extraction-based” method (Smith and Brady, 1995), which extracted a certain object by image processing techniques, such as edge detection and object segmentation processing. However, the appearance of

human body in video sequences may not be very concrete and is easily corrupted by noise. This approach suffers from lack of robustness to lighting, nature of the background, and occlusions.

To make algorithms more robust, different kinds of video representations were introduced to capture the invariance in the video, such as local image features or spatio-temporal interest points in [Laptev and Kindeberg \(2003\)](#), which provided a compact and abstract representation for patterns within a given image. The applications included object detection, tracking, and segmentation. The performance was demonstrated to be robust for variations of background. The so-called Scale Invariant Feature Transform (SIFT) points were then proposed in [Lowe \(2004\)](#), and a method was designed for extracting distinctive invariant features based on the SIFT points. The SIFT points were selected by calculating the Difference of Gaussians at every pixel and representing the descriptors in different directions. Points of interest can also be encoded as a histogram ([Schuldt et al., 2004](#)) and this kind of representation is combined with a Support Vector Machine (SVM) ([Burges, 1998](#)) or some other probabilistic model. Using similar ideas, a generative graphical model in [Niebles et al. \(2006\)](#) used the interest points for human action recognition. This method analyzed the human action directly in the space-time volume without explicit motion estimation ([Shechtman and Irani, 2005](#)).

On the action understanding side, people focused on detecting the type of action by motion analysis. After extracting the human in the video clip, the human can be represented by several special parts, such as arms and legs. The action is analyzed by detecting the motion of these parts and models for different actions can be learned from the given motions during the training process ([Yang et al., 2009](#)). With the various backgrounds and different viewing angles, how to effectively detect the critical points on human body becomes the main difficulty for these approaches.

To avoid the detecting difficulties in critical points, an appearance-based approach is proposed to solve more general problems. Subspaces can be learned from the training video clips and used to model the query ones. Traditional subspace modeling includes Principle Component Analysis (PCA) ([Turk and Pentland, 1991](#)), Linear Discrimination Analysis (LDA) ([Belhumeur et al., 1997](#)) and so on. Linear Projection Preserving is proposed in [He and Niyogi \(2003\)](#) to build a graph on understanding the neighboring information.

Another popular approach is to treat the human action as a sequence and learn the model from the difference in the temporal domain. Besides, in [Hastie and Stuetzle \(1989\)](#), a non-linear principal curve approximation was developed. Intuitively, it is a curve passing through the center of the data points cloud, with a smoothness constraint. In [Kegl et al. \(2000, 2002\)](#), it was demonstrated that as long as the second moment of the data points cloud is finite, there must be a principal curve, and an iterative polygonal principal curve learning algorithm was developed.

In this work, we model human action video clips as manifolds in the scaled appearance space over time. Video clips of different human actions performed by different subjects under different image formation conditions span a space with complex structure and relationships. By scaling the original video frames into icon images, the local noise can be effectively attenuated while the information about the action is maintained. The formulation can be divided into video representation, subspace learning, and matching problem.

3.1. Video representation

Video representation is the first and one of the most important sub-problems in video pre-processing. A good representation should include the key point and useful information for discrimination while discarding unnecessary information.

Generally, in video processing, video frames are usually represented as a matrix. In our method, we use the luminance information to keep the data in every single frame, with a vector structure. To simplify pre-processing, pixel values are directly extracted as features. The video frame is first down-sampled to a smaller icon to reduce spatial redundancy together with noises, and then the icon is projected into a high dimensional space and become a point. In this way, different video clips of different human actions performed by different subjects under different image formation conditions span a space with complex structure and relationships. The spatial features in the clips are kept in a vector form while the temporal ones are included in the trajectory as well.

Considering a video clip which contains n frames, with $W \times H$ pixels in each frame, the k th frame F_k can be represented as a point in the space $\Re^{W \times H}$. Actually the frame of size $W \times H$ still contains more information than necessary, so down-sampling it will reduce the number of elements while keeping adequate information for recognition. The down-sampling step reduces the original frame down to a smaller $w \times h$ one. By down-sampling each frame can be further represented as a point in a space of smaller dimension, i.e., $\Re^{w \times h}$. After this processing, the trajectory still contains sufficient statistical discriminative information for classification.

3.2. Dimensionality reduction

To further simplify the processing, another pre-processing step is introduced by subspace learning. In this step a global subspace is learned. By projecting every sample point to the subspace, the discriminative information in the set is maintained while the number of dimensions is reduced for faster processing. A global PCA ([Turk and Pentland, 1991](#)) is applied here to reduce the dimensionality of the space. Consider an n -frame video sequence; given a frame $F_k \in \Re^{w \times h}, k \in [1, n]$, the subspace learning can be expressed as:

$$x_k = AF_k = [a_1, a_2, \dots, a_{w \times h}]F_k, a_j \in \Re^d, \quad (1)$$

where the subspace projection A , of size d by $w \times h$, is obtained from an unsupervised local learning, with the objective of preserving the maximum amount of information, while keeping the number of dimension at an acceptable level. Each a_j is a $d \times 1$ column vector of matrix A .

[Figure 1](#) shows three groups of curves, each for a different human action in the Cambridge hand gesture dataset in 3-D space. In the figure, video clips containing different actions have different shapes, and as one can judge some actions are clearly different, while for others it is rather difficult to distinguish them since the 3-D view cannot offer enough visual information for doing so. Actually the geometry information of these curves already contains sufficient statistics to recognize the different human actions. In the next three sections, we will propose our approach based on the statistical information.

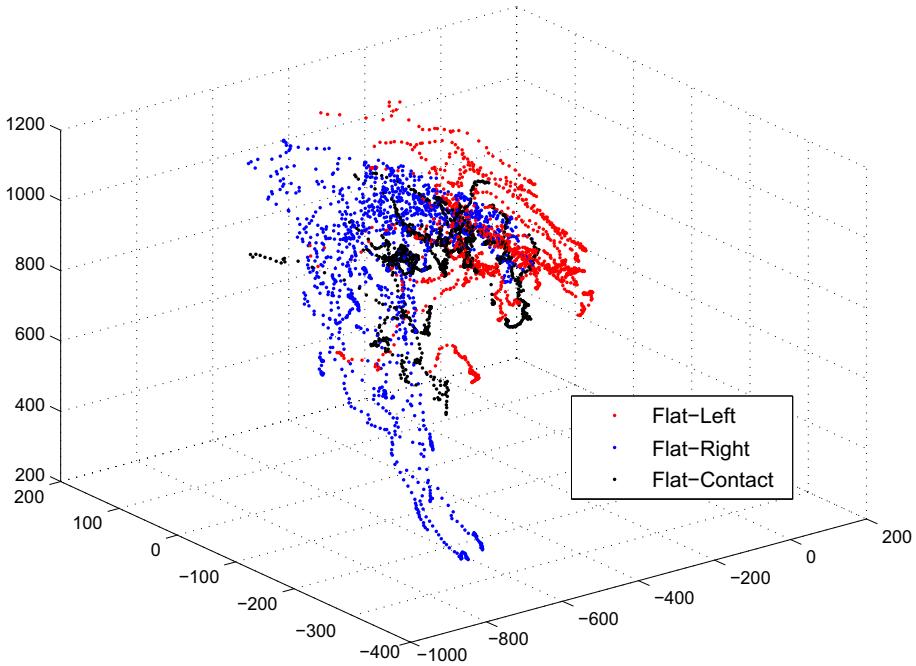


Fig. 1. Example trajectories for three hand gestures.

3.3. Maximum likelihood detection

Each video frame is represented as a point in a high dimensional space R^d after the pre-processing and dimension reduction, so that each video clip is a trajectory in this space. The representation is still not simple enough to discriminate the different action classes.

Assume that the training frame set $\{x_1, x_2, \dots, x_n\}$ in the R^d space is with Gaussian distribution, then we can obtain the mean m_x and the variance σ_x , respectively. Given a query frame q and a training frame x , the likelihood that q and x have same action label is also under a Gaussian distribution, i.e.,

$$\mathcal{L}(q; x) \sim N(m_x, \sigma_x). \quad (2)$$

Since x is a Gaussian Mixture, the likelihood can be further computed as

$$\mathcal{L}(q; x) = \mathcal{L}(q; m_x, \sigma_x) = \frac{1}{2\pi^{d/2}\sigma_x^{1/2}} e^{-\frac{1}{2}(q-m_x)^T \sigma_x^{-1} (q-m_x)}. \quad (3)$$

The frame likelihood is defined as Eq. (3), and then we develop the trajectory likelihood. Given a query trajectory $q(t)$, composed by t frames $\{q_1, q_2, \dots, q_t\}$ and

a training sequence $x(t)$, the likelihood between $q(t)$ and $x(t)$ can be computed as:

$$\mathcal{L}(q(t); x(t)) = \prod_{t=1}^n \mathcal{L}(q(t); m_x(t), \sigma_x(t)). \quad (4)$$

If there are totally k training trajectories, we can compute the k likelihood and then decide the action label of $q(t)$ by using Maximum Likelihood decision as follow,

$$k^* = \operatorname{argmax}_k \mathcal{L}(q(t); x_k(t)). \quad (5)$$

However, in the trajectory matching, another challenge is to match up the sequences with different durations. Considering an n -frame sequence, which can be represented as an n -point trajectory, is denoted as $\{x_1, x_2, \dots, x_n\}$. Another m -frame sequence, denoted as $\{y_1, y_2, \dots, y_m\}$. There can be multiple matching options. In the matching process, trajectories with different length may have different matching offset. If we consider every possible matching option, the growth in difference of duration will make the matching complexity grow exponentially, which is computationally prohibitive. In this work we compute point-to-point likelihood, which strictly keeps the temporal information in the trajectory. It is constrained that points must be matched according to order. No skipping, repeating, or crossing is allowed for matching. Therefore, given the query and training trajectories with durations t_q and t_x , the likelihood computation is corrected into

$$\mathcal{L}(q(t_q); x(t_x)) = \begin{cases} \min_h \prod_{t=1}^n \mathcal{L}(q(t_x + h); x(t_x)), & t_q > t_x, \\ \prod_{t=1}^n \mathcal{L}(q(t_q); x(t_x)), & t_q = t_x, \\ \min_h \prod_{t=1}^n \mathcal{L}(q(t_q); x(t_q + h)), & t_q < t_x. \end{cases} \quad (6)$$

In this way, a time align process is performed and the matching is simplified into a linear level. In order to find the optimal matching, likelihood of all the possible matching offset between two trajectories are computed. The maximal value is selected as the best evaluation. Then the decision can be made as,

$$k^* = \operatorname{argmax}_k \prod_{t=1}^n \mathcal{L}(q(t_q); x_k(t_{x_k})). \quad (7)$$

3.4. Luminance aligned projection distance approach

The Bayesian-based likelihood solves the trajectory matching problem effectively, but not efficiently. The computation for likelihood in Eq. (3) is very complicated and limits the method to many real-time applications. In this section we propose a simplified version, the Luminance Aligned Projection Distance (LAPD) approach, to efficiently solve the matching problem.

Given a training set after dimension reduction, the parameters d, m_x , and σ_x are constant for every query clip. So the computation of likelihood can be simplified by removing the first multiplier factor in Eq. (3), i.e.,

$$\mathcal{L}(q; x) = \mathcal{L}(q; m_x, \sigma_x) = e^{-\frac{1}{2}(q-m_x)^T \sigma_x^{-1} (q-m_x)}. \quad (8)$$

The exponential function is a monotone increasing function, and the detection of maximum likelihood is equalized to finding the minimum Mahalanobis distance from query clip q to m_x . The distance measurement is more reliable when a subject is repeating same action under illumination conditions or in different background. Especially, the subspace can be directly obtained by decomposing the covariance matrix, $S = \sigma^{-1}\sigma = A^T A$. Based on this observation, instead of computing probabilities, we propose a distance-based approach to detect the maximum likelihood. Intuitively, the distance between two trajectories is believed to be an effective and reliable measurement for similarity. Samples which have similar content should have smaller distance, as compared to those with dissimilar content.

Basically, the square of distance between two points in the subspace A can be defined as Eq. (9).

$$d(x, y) = \|A(x - y)\|^2 = (x - y)^T A^T A (x - y), \quad (9)$$

where both x and y are points in \Re^d . Specially, when A is a unit matrix, $d(x, y)$ is the Euclidean distance. Also, when A is the variance in the Gaussian training set, the distance definition becomes a special variation of the likelihood defined in Eq. (3). In our proposed solution, the subspace metric A is computed by either PCA for unsupervised learning or LDA for supervised learning.

Consider a point x and a trajectory Y composed of a group of points $\{y_1, y_2, \dots, y_n\}$. The square of distance from a point x to the trajectory can be defined as the minimal point-to-point distance, i.e.,

$$d(x, Y) = \min_i d(x, y_i) = \min_i \|A(x - y_i)\|^2. \quad (10)$$

Furthermore, to compute an effective distance which gives reliable similarity representation, the Luminance Aligned Projection Distance (LAPD) is proposed based on the maximum likelihood formulation: given a pair of trajectories, finding an optimal matching offset h in the longer trajectory started where the afterward average point-to-point distance is minimized.

Suppose trajectories are denoted as $x_{j,k}(t)$, for curve j belonging to action class k , and for each class, there are $j = 1, \dots, n_k$ curves, t is the frame index which varies from 1 to n . Then for an unknown video clip trajectory $y(t)$ with m frames, and a known action video clip $x(t)$ of n frames, assuming $m < n$, the LAPD between x and y is defined in Eq. (11).

$$d_{\text{LAPD}}(x, y) = \min_h \frac{1}{m} \sum_{t=0}^{m-1} \|A(x(t+h) - y(t))\|^2. \quad (11)$$

Let us assume that we have K action classes and each has $j = 1, \dots, L$ training clips. We denote by $x_j^k(t)$ the j th training clip belonging to class k . In this way, multiple models are constructed for each action class. Then for an unknown clip $y(t)$, recognition can be implemented based on the minimum LAPD,

$$k^* = \operatorname{argmax}_k \min_j d_{\text{LAPD}}(y(t), x_j^k(t)). \quad (12)$$

For each incoming query video clip C , we calculate the LAPD between C and each training trajectory from different action classes. With multiple trajectory models in the training set, the variation of subject or background in the query video clip is much easier to handle. A LAPD distance array is generated based on Eq. (11). After sorting the entries of the distance array, the M smallest values are selected and the corresponding training action labels are recorded. Given the first M labels, voting is applied to count the number of labels for each action class. The final decision is based on the label with the most votes.

In this method, we focused on finding the relationship between the two trajectories in subspace. Instead of computing the distance directly, a best matching point is firstly found by trying every possible offset in matching the two trajectories which minimize the distance between them. The spatial information is maintained in the trajectory coordination, while the temporal features are kept by continuous point-to-point matching. This processing removes the effect brought by noise, and proved to be robust against some other factors such as scaling and background changes.

3.5. Differential luminance field trajectory approach

The computation of distance in previous section required less effort than likelihood, but still complicated. Consider the training clip with n frames, and there are h possible offset, a query clip will compute $n \times h$ point-to-point distance to get one LAPD, and totally the complexity is approximately $O(nhN)$, where N is the number of clips in the training set. In this section we derive a faster differential-based algorithm, Differential Luminance Field Trajectory (DLFT).

Spatio-temporal features are extracted from the pre-processed data in the following ways: the spatial features are firstly extracted from the appearance of frames. Then a differential is applied on neighbor points of the trajectories to maintain the temporal features. In this step, the following signal is formed.

$$dx_k = \begin{cases} 0, & k = 1, \\ |x_k - x_{k-1}|, & \text{otherwise.} \end{cases} \quad (13)$$

For the first frame, we skip capturing the differential information and set it to zero since there is no previous frame. From the second frame on, we record the difference between neighbor frames as a sequence signature. The information included in the signature is dominant for discrimination, but in a quite simple format. Each n -frame sequence can be represented as a group of differential signals, $\{dx_1, dx_2, \dots, dx_n\}$ in this way.

For a real-time application, the durations of video clips, n , are different from each other, which means trajectories are of different length. In DLFT, to solve this problem and better utilize the features, a transformation is applied on the differential trajectory to normalize the length of the data to represent each video clip while maintaining the useful information. A 64-point DCT or DFT is performed on a sliding time window over the differential trace, as shown in Eq. (14).

$$dx_k \leftrightarrow \Gamma(dx_k), \quad (14)$$

where the $\Gamma(\cdot)$ is the DCT or DFT transformation. Trajectories with any length, $\{dx_1, dx_2, \dots, dx_n\}$, is uniformly converted into a 64-point vector. Noticed that for DFT, which is not a real transform, only the magnitude of the signal is utilized as feature representation. The signals in frequency domain are extracted as features and utilized for further processing. The transformation will probably introduce one or two seconds initial delay which should be acceptable for in most real-time or on-line applications. Video sequences are equalized to same length via the transformation. After the transformation, a frequency domain subspace is obtained for discrimination and then KNN classifier can be used for a final decision.

Given a query video clip q , the objective is to find a clip k in the training set which is closest to q , and label q with the label of k , i.e.,

$$q^* = \underset{dx_k}{\operatorname{argmax}} \|A(\Gamma(dx_q) - \Gamma(dx_k))\|^2, \quad (15)$$

where A is learned from the frequency domain, and dx_q and dx_k are the differential features of query and training, respectively.

In this method, the computational complexity is greatly reduced: after processed by Eqs. (13) and (14), a query will be compared with N clips in the training set only once to get their distance. Generally we just use 32-point or 64-point transformation, so the complexity is improved to $O(N)$. The processing just keep the spatial information for the first frame and uses differential processing after that. Many details in the video are lost, which may degrade the performance on recognition accuracy. There is a tradeoff between the performance and computation complexity.

4. DLFT and LAPD solutions

In this section we introduce our solution of DLFT and LAPD separately. The KTH data set, which is tested in [Laptev and Kindeberg \(2003\)](#) and [Schuldt et al. \(2004\)](#), is used as an example to illustrate the proposed methods in detail.

After pre-processing by down-sampling and PCA dimension reduction, some 2-D luminance field trajectories are plotted in the Fig. 3 as example trajectories. The ones contain the “hand action,” i.e., *boxing*, *handclapping*, and *handwaving*, are plotted in blue, while the others containing the “body action,” i.e., *jogging*, *running*, and *walking*, are plotted in red. From this 2-D figure we can intuitively distinguish the hand action and body action by the shape of the curves. These curves will be further demonstrated to have enough statistical information for recognition tasks.

4.1. DLFT solutions

In DLFT solution the differential values in the trajectory are computed according to Eq. (13), each video clip is represented as a vector with different length. Both the spatial and temporal information is maintained in this step: the spatial information

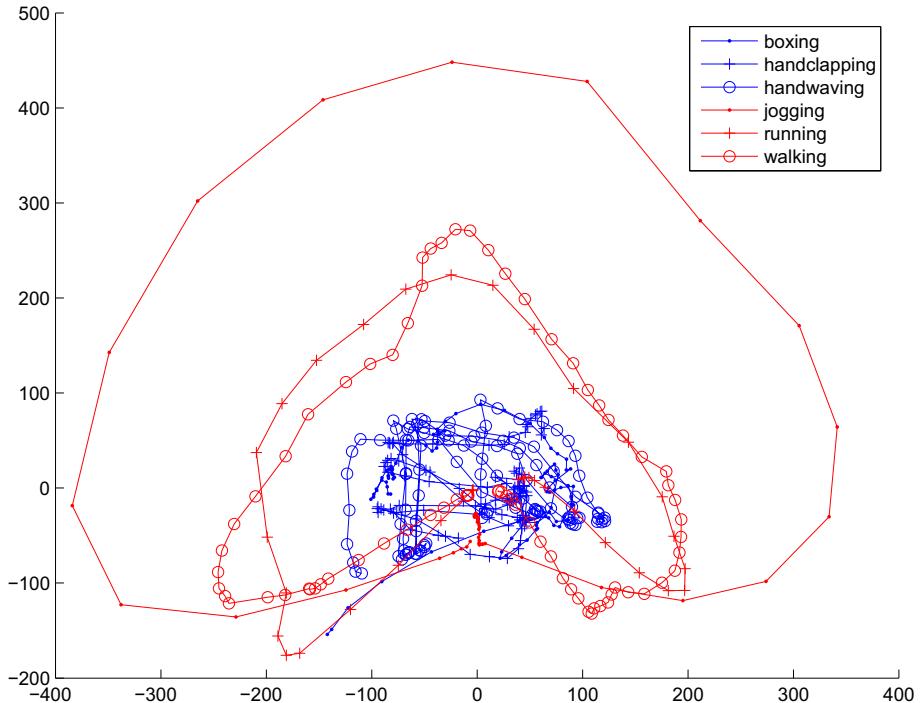


Fig. 2. 2-D human action luminance field trajectory examples.

is kept in the value of signal while the temporal information is represented as the relationship between neighboring points. An example of the differential trace is plotted in Fig. 3. Although the shape looks different due to the different durations of sequence, it is still difficult to judge the pattern.

Then a transformation is applied to align all clips into equal length. In this work we apply DCT and DFT separately, and the sequence after transformation is plotted in Fig. 4. The first several DCT coefficients are of different pattern for different kind of actions, while the high-frequency coefficients are almost zero. In DFT there are also some dominant coefficients on different positions. The certain pattern can already be understood in this figure.

Standard supervised learning techniques like LDA in Fisherface (Belhumeur et al., 1997), as well as graph embedding-based Locality Preserving Projection (LPP) (He and Niyogi, 2003) are applied to these feature vectors in the DCT or DFT domain.

The classification is based on a simple K-Nearest Neighbor (KNN) classifier. The training clips projected in LPP or LDA space are modeled as a six-class Gaussian mixture, and the classification is done by assigning maximum likelihood action labels. Experimental results and discussions are presented in Section 5.

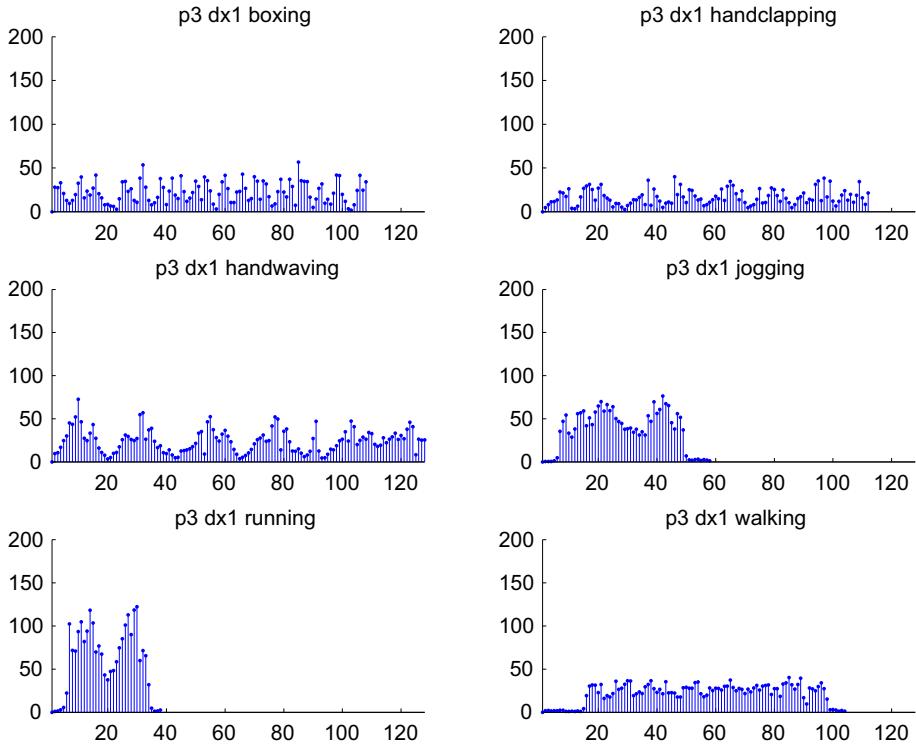


Fig. 3. Examples for differential trace for different actions.

4.2. LAPD solutions

In the LAPD processing we focused on the computation of inter-trajectory distance. The trajectories are the same as plotted in Fig. 2.

In this solution, given a query clip, the distance between the query and each training trajectory is pairwise computed by Eq. (11), and a distance array is generated for each query. The distance from the query to each action category is computed and a histogram is shown in Fig. 5. The decision is made by selecting the smallest distance, which is quite obvious in the figure.

5. Experiments

We have tested our methods on three different datasets, the *KTH human action dataset*, the *Cambridge Hand Gesture dataset*, and the *Youtube action dataset* for human action recognition and hand gesture recognition separately. These datasets cover variations in appearance, illumination, background, and spatio-temporal cues. Besides proposed approaches, we also tested the DTW method on the dataset above, and compare it with some results obtained by other approaches in the recent literature.

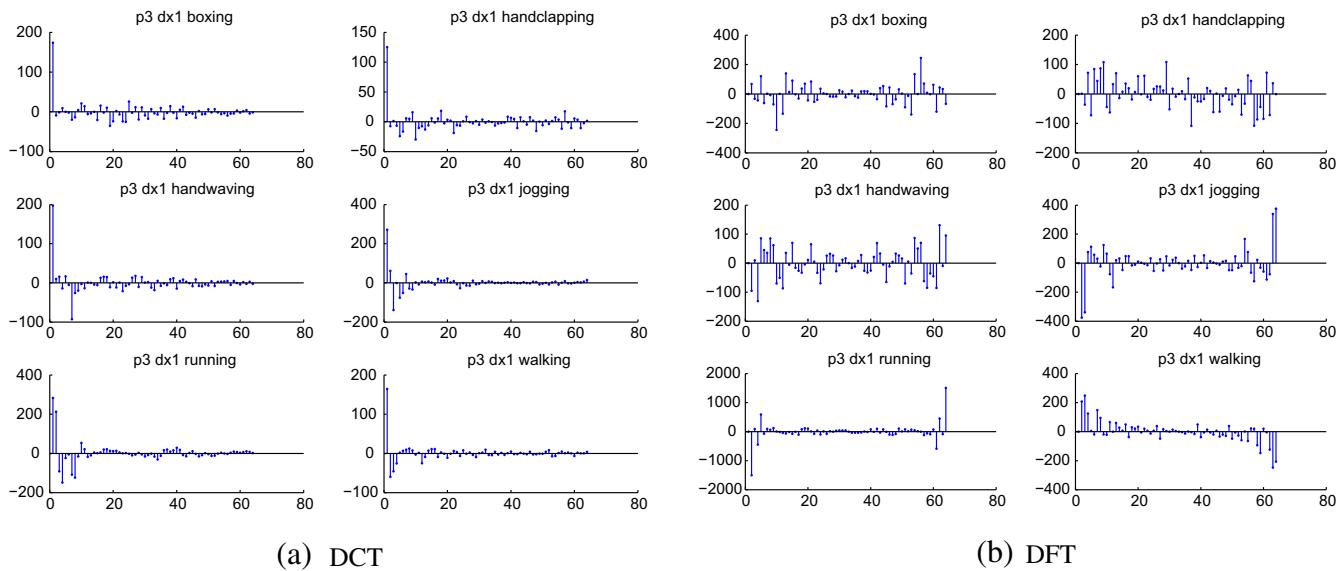


Fig. 4. Transformed value of differential traces (a) by DCT, (b) by DFT.

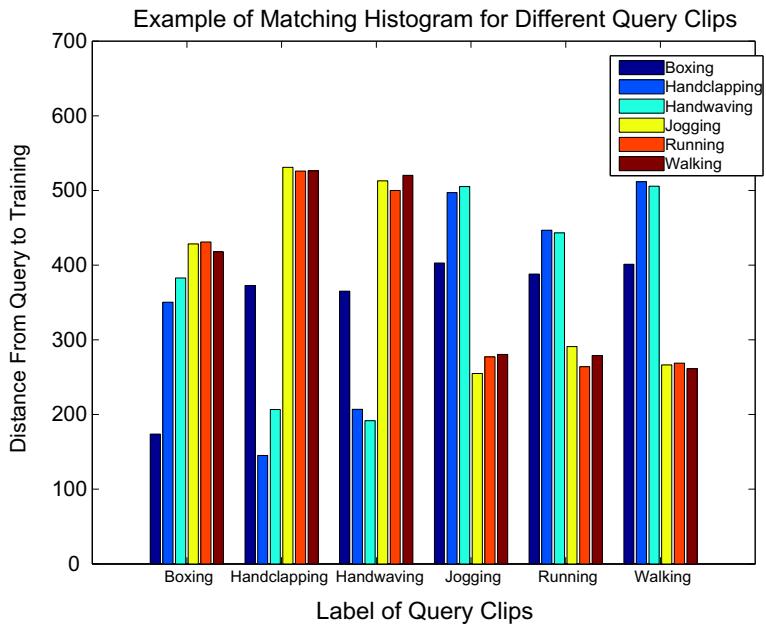


Fig. 5. LAPD matching example in KTH dataset.



Fig. 6. KTH human action clip example.

For all the datasets, our implementation is based on the leave-one-actor-out setting, where the classifier is trained using all video sequences except those corresponding to the actor in the test video. This processing is repeated many times until each video has been treated as the test video (see Fig. 6).

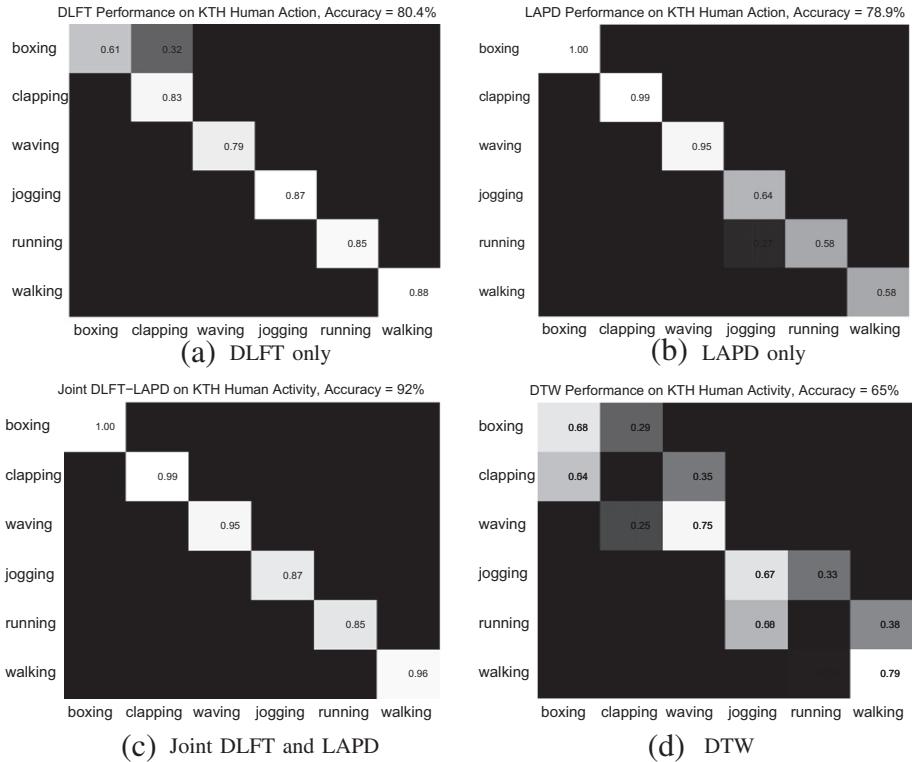


Fig. 7. Performance on KTH human action recognition (a) by DLFT only, (b) by LAPD only, (c) joint DLFT and LAPD, (d) DTW.

5.1. KTH human action dataset

5.1.1. Dataset introduction

To test the developed algorithm, we use the human activity data set from Schuld et al. (2004), which contains six human actions, “*boxing*,” “*handclapping*,” “*handwaving*,” “*jogging*,” “*running*,” and “*walking*.” Actions are performed by a total of 25 subjects in four different settings:

S1: outdoors; *S2*: outdoors, with camera zooming; *S3*: outdoors, with different clothes on; *S4*: indoor.

For each setting, each action has four video clips, with each segment’s start and end frame number listed as a ground truth file. Each setting will have $4 \times 25 \times 6 = 600$ actions, and the data set comprises of a total of 2391 clips, with a small number of entries missing.

The video clips are of 160×120 pixel resolution, and in processing stage, we down convert the sequence into 20×15 icon image sequences for trajectory modeling. Some examples from the action clips in Schuld et al. (2004) are plotted in Fig. 7. From left to right, the top row actions are walking, jogging, and running, and the bottom row actions are boxing, hand waving, and hand clapping.

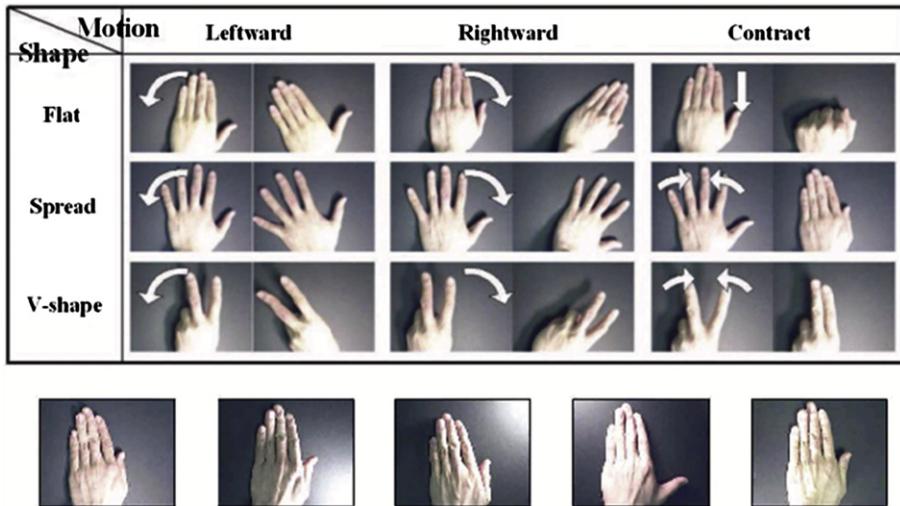


Fig. 8. Hand gesture database: (top) nine gesture classes by three motion directions and three hand shapes; (bottom) five different backgrounds with different illuminations.

5.1.2. Simulation result

In the pre-processing stage, we down-sampled the original video frame to 20×15 icons and applied a global PCA to reduce the number of dimension to 64. Thus the video sequences are represented by trajectories in \mathbb{R}^{64} and the differential trajectories are computed. To deal with the different durations of video clips and to eliminate the high-frequency coefficients, a 64-point DFT is taken. Then LDA supervised learning is applied on the differential signals and KNN classifier is used to assign a label for the test video sequence. The confusion map is shown in Fig. 7a, and the overall recognition accuracy is 80.3%.

For the LAPD approach, the 64-dimensional feature is used again after pre-processing for dimensionality reduction. Supervised learning is later applied to discriminate the different patterns and to further reduce the number of dimension for easier LAPD computation. The result is shown in Fig. 7b, with an overall accuracy of 78.9%.

From the result of the proposed two approaches, it is observed that DLFT has better performance on behaviors containing more temporal information, such as *running* and *jogging*, while LAPD has advantages in appearance-based behaviors like *handwaving* and *handclapping*. Intuitively the performance can be further improved by combining the strong points of both methods. During the simulation it is found that there is little motion in some action classes, such as *boxing*, *handwaving*, and *clapping*, which is referred as static classes. On the other hand, a lot of motion can be detected from the *jogging*, *running*, and *walking* sequences, named as motion classes. The value of differential trajectory of static classes is closer to zero. Theoretically, the DCT and DFT do not have good discriminative performance on such sequences, so the recognition result from DLFT in these classes is not as good as the corresponding result of LAPD. On the other hand, for motion classes, the differential

signal is much more discriminative, which provides better performance than LAPD. A combination scenario is therefore implemented based on this observation.

The combination is processed with the following steps. In the feature extraction step, both the luminance trajectories and their differential values are stored. Then the strength of the temporal information is evaluated by measuring the sparseness of the differential signals. A threshold is set to discriminate whether the temporal features are stronger than the spatial ones or not. When there is stronger temporal information, it means there is large motion in the video sequence and DLFT is applied to classify the query sequence into the last three actions, i.e., *jogging*, *running*, and *walking*. Otherwise, LAPD is applied to utilize the appearance features for discrimination among *boxing*, *handwaving*, and *clapping*. The result of the combination of LAPD and DLFT is shown in Fig. 7c with a final accuracy of 92%.

For better comparison, the DTW algorithm is also implemented in our experiment. With the same pre-processing, the DTW is applied directly on the trajectory in \mathbb{R}^d . An optimal alignment can be found between two trajectories and the point-to-point Euclidean distance is computed as a measurement of similarity. The label decision is done with a KNN classifier and the result is shown as in Fig. 7d. The overall accuracy is 65%.

DTW is a method with time warping component with flexibility. However in the trajectory matching, it is restricted, the first point from one trajectory must be matched with the first point of each other trajectory, and the same for the end point. We refer this restriction as a “begin/end curse.” The flexibility only exists in the middle of trajectories. In our action recognition problem, the first and last frame of sequences from same action label is not necessary to be similar. The begin/end curse degrade the recognition accuracy drastically.

5.2. Cambridge hand gesture dataset

5.2.1. Dataset introduction

To demonstrate the robustness and versatility of the algorithm, we also tested another dataset, the *Cambridge Hand Gesture Dataset*, which is composed of 900 image sequences with nine different hand gesture classes (Kim et al., 2007a). These classes are defined by three primitive hand shape: *Flat* (*F*), *Spread* (*S*), and *V-shape* (*V*), and three primitive motion directions: *Leftward* (*L*), *Rightward* (*R*), and *Contract* (*C*). There are totally nine gesture classes by combining the two factors above. Each class contains 100 image sequences, with five different illumination cases. An example is shown in Fig. 8.

Each sequence in the dataset has a different number of images, varying from 37 to 119, and the total number of image is 63,188. The original image is a 320×240 color image, and we firstly reduce the number of data by converting it to gray image and then down-sample it to a 32×24 icon. We then process these icons with the DLFT and LAPD approach, respectively.

5.2.2. Simulation result

The hand is composed of many connected parts and the motion is highly articulated. Without prior information, it is difficult to guess what kind of appearance is

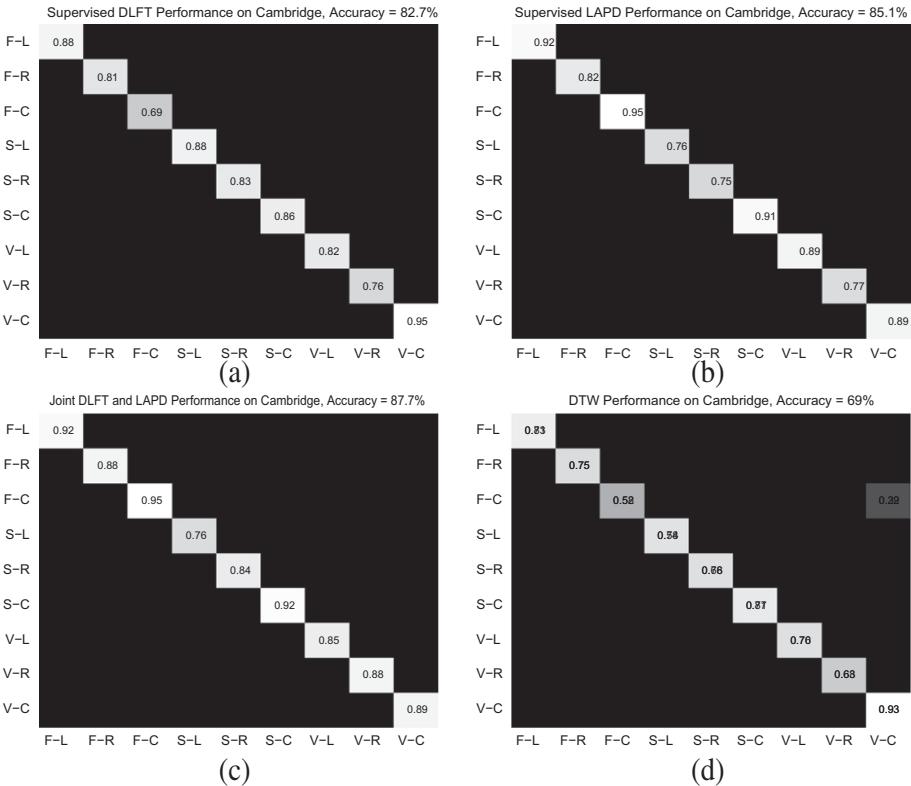


Fig. 9. Performance on cambridge hand gesture recognition (a) by DLFT, (b) by LAPD, (c) by Joint DLFT + LAPD, (d) by DTW.

contained in the image even if it is known to be a hand. To better discriminate the appearance, we keep 32×24 pixels as icons in the pre-processing. For a global dimensionality reduction, a PCA is applied on $\mathcal{R}^{32 \times 24}$ and 64-dimentional trajectories are treated as representations for the image sequences for further processing.

With the DLFT approach, a 64-point DFT is applied directly on the trajectory to solve the duration problem. The simulation result is shown in Fig. 9a, the overall recognition accuracy is 82.7%.

In the LAPD approach, the aligned distance is computed and the label of the query clip can be decided by a KNN classifier. The recognition accuracy is 85.1%. Figure 9b shows the confusion map for gesture recognition. For the individual set testing, the comparison results are listed in Table 1. The numerical result is comparative to the one in Kim et al. (2007a) and better than some other results reported in the literature Yuan et al. (2010).

In order to better utilize the correlation between spatial and temporal features in the video sequence, a joint of DLFT and LAPD is applied for hand gesture recognition. The result is shown in Fig. 9c, with an accuracy of 87.7%.

Table 1
Hand gesture recognition accuracy %

Method	Set1	Set2	Set3	Set4	Set5	Average
DLFT (unsupervised)	76	75	68	74	72	73.0
LAPD (unsupervised)	83	81	77	80	79	80.0
DLFT (supervised)	84	81	86	84	78	82.7
LAPD (supervised)	91	85	78	84	87	85.1
TCCA (Kim et al., 2007a)	81	81	78	86	—	81.5
Nieble (Niebles et al., 2006)	70	57	68	71	—	66

Dynamic Time Warping has also been tested on the hand gesture dataset and the result is shown in Fig. 9d. The accuracy is not as high as with LAPD, since even in the same hand motion pattern, the magnitude of motion may be different with different people.

5.2.3. Analysis

The object of appearance information in the hand gesture data set is mainly the hand, and the change in the background is a factor to test the robustness of any algorithm. From our KNN result, we found out that in the correct cases, the nearest trajectory in the training set is always in the same class and same background as the query clip. The effect of the changing of the illumination of the background will not influence the recognition performance.

As shown in both Kim et al. (2007a) and our LAPD solution, there are several confusions between the class *spread* and *flat*, for either left or right direction. These are mainly due to very little difference in appearance, and the details are lost when sampling the original frames down to a small icon. The DLFT approach, however, is not suffering from this problem, because the differential operation cares more about the temporal properties, compared with the spatial one. Therefore, a joint of the DLFT and LAPD can take advantage from both spatial domain and temporal domain, and the recognition accuracy is better than the separate method.

5.3. Youtube action dataset

5.3.1. Dataset introduction

A more challenge dataset we used to test our proposed solution is the *Youtube human action dataset* (Liu et al., 2009). eleven different actions, *Basketball shooting*, *Cycling*, *Golf Swing*, *Diving*, *Horse Riding*, *Soccer Jungling*, *Swinging*, *Tennis Swinging*, *Trampoline Jumping*, *Volleyball Spiking*, and *Walking with a Dog*, are included in the dataset. Each action is repeated by 25 different persons for several times and totally 1577 video clips are involved. Example frames are shown in Fig. 10.

5.3.2. Simulation result

This is a challenge human action dataset because of large variations in camera motion, object appearance and pose, object scaling, viewpoint, and very complicated



Fig. 10. Example frames of Youtube action dataset.

background. Therefore, in our DLFT approach, we applied a 64-point DCT instead of the DFT in the proposed algorithm. The result of the DLFT method on the Youtube database is shown in Fig. 11a. The recognition accuracy is 71.7% for all 11 actions.

The first 64-dimensional feature was kept in the LAPD and DTW experiment. A global subspace is learned with the training set. The results are shown in Fig. 11b and c, respectively. LAPD results in an accuracy of 91% and DTW in about 65%.

Since there are very complicated background variation in the video sequence, the temporal features do not contribute as much as spatial one. The differential processing, which will theoretically cancel some of the background effect, cannot differentiate the human with background properly. This cause the differential signal less meaningful, especially for static classes. The result from our proposed DLFT method is comparable with the ones in the literature such as Liu et al. (2009) and Liu et al. (2009), but not as good as LAPD. On the other hand, LAPD accurately describe the similarity between similar videos. The statistical information in appearance is highly utilized by LAPD solution, and query trajectories are usually matched with a training trajectory with similar appearance.

5.4. Discussion

The proposed DLFT and LAPD approaches are mainly aiming at preserving both spatial and temporal features, which are crucial for action recognition in video sequences. In the KTH dataset, the six classes of human action have different spatio-temporal content: the three-hand actions have more spatial information than the temporal one, while the body action mainly consists of temporal features.

The DLFT is mainly focused on the difference between neighboring frames by calculating the differential value. The statistical feature, which is used for classification, is the variation distribution of differences in the video sequence. This feature does not consume many resources due to the limited number of video frames, and can be used for on-line applications. It is also theoretically and practically

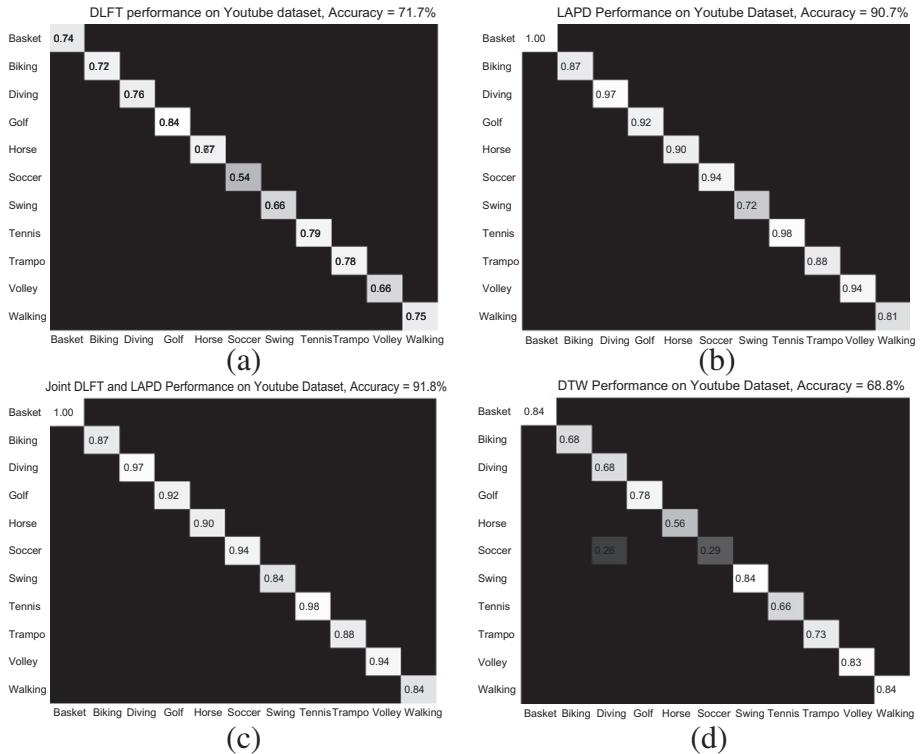


Fig. 11. Simulation result for Youtube Action dataset: (a) DLFT approach, (b) LAPD approach, (c) Joint DLFT-LAPD approach, (d) DTW algorithm.

robust to the change of background or occlusions, because after taking differences the unstable factors above are all removed.

For LAPD, the dominant feature for recognition is the distribution of frames in the transformation domain. The aligned method is designed to address the difference in video durations. As is proved in the experiments the performance is quite accurate, even with a small number of training samples.

Both methods above still have a common advantage: they are both content independent and only very simple pre-processing is needed. They can be applied to both human action recognition and hand gesture recognition. Theoretically the algorithm itself is not dependent on the video content, and for every possible video sequence, the method can offer good recognition accuracy as long as the classes are defined clearly.

6. Conclusion

In this work, we proposed two approaches for video content recognition without object level learning. The DLFT solution is computationally efficient. The major time cost is brought by the use of the DCT or DFT. Generally, such one or two

seconds can be acceptable for most real-time applications, especially large-scale scenarios.

Another solution we proposed is to utilizing aligned projection distance approach. We still use the trajectory to represent the video clip and to calculate the distance between every two clips to define how similar they are to each other. The results show that the recognition accuracy is comparable or better than other techniques in the literature. Regarding complexity, the off-line generation of the distance matrix costs some time, and it will also increase with the size of the training set. However, the on-line recognition for the query is very fast and is also suitable for most applications when the training set is fixed.

Further work includes a potential combination of the two proposed solutions. A tradeoff needs to be balanced in between complexity and performance. Further optimization may help the aligned projection to have a higher efficiency.

References

- Aggarwal, J.K., Cai, Q., 1999. Human motion analysis: a Review. *Comput. Image Understand* 73 (3).
- Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J., 1997. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* 19 (7), 711–720.
- Bobick, A.F., Davis, J., 1996. An appearance-based representation of action. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Bobick, A.F., Davis, J., 2001. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (3).
- Burges, C., 1998. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Disc.* 2.
- Cedras, C., Shah, M., 1995. Motion based recognition: a survey. *Image Vis. Comput.* 13.
- Chang, S.F., Chen, W., Meng, H.J., Sundaram, H., Zhong, D., 1998. A fully automated content-based video search engine supporting spatio-temporal queries. *IEEE Trans. Circ. Syst. Video Tech.* 8 (5), 602–615.
- Cherla, S., Kulkami, K., Kale, A., Ramasubramanian, V., 2008. Towards fast, view-invariant human action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops.
- Efros, A., Berg, A., Mori, G., Malik, J., 2003. Recognition action at a distance. In: IEEE International Conference on Computer Vision.
- Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R., 2007. Actions as space-time shapes. In: IEEE Trans. Pattern Anal. Mach. Intell. 29 (12), 2247–2253.
- Hastie, T., Stuetzle, W., 1989. Principal curves. *J. Amer. Stat. Assoc.* 84, 502–516.
- He, X., Niyogi, P., 2003. Locality preserving projections. In: Advances in Neural Information Processing Systems. Vancouver, Canada.
- Hoey, J., Little, J., 2000. Representation and recognition of complex human motion. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Ke, Y., Sukthankar, R., Hebert, M., 2005. Efficient visual event detection using volumetric features. In: IEEE International Conference on Computer Vision, pp. 166–173.
- Kegl, B., Krzyzak, A., Linder, T., Zeger, K., 2000. Learning and design of principal curves. *IEEE Trans. Pattern Anal. Mach. Intell.* 22 (3), 281–297.
- Kegl, B., Krzyzak, A., 2002. Piecewise linear skeletonization using principal curves. *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (1), 59–74.
- Kim, T.Y., Wong, S.F., Cipolla, R., 2007a. Tensor canonical correlation analysis for action classification. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Kim, T.K., Kittler, J., Cipolla, R., 2007b. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (6).

- Laptev, I., Kindeberg, T., 2003. Space-time interest points. In: IEEE International Conference on Computer Vision, Nice, France, pp. 432–439.
- Laptev, I., Marszałek, M., Schmid, C., Rozenfeld, B., 2008. Learning human actions from movies. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Li, Z., Fu, Y., Yan, S., Huang, T.S., 2008. Real-time human action recognition by luminance field trajectory analysis. In: ACM Multimedia.
- Liu, J., Luo, J., Shah, M., 2009. Recognizing realistic actions from videos “in the wild.” In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).
- Liu, J., Yang, Y., Shah, M., 2009. Learning semantic visual vocabularies using diffusion distance. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).
- Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* 60 (2), 91–110.
- Niebles, J.C., Wang, H., Li, F., 2006. Unsupervised learning of human action categories using spatial-temporal words. In: British Machine Vision Conference.
- Oikonomopoulous, A., Patras, I., Pantric, M., 2005. Spatiotemporal saliency for human action recognition. In: IEEE International Conference on Multimedia and Expo.
- Pierobon, M., Marcon, M., Sarti, A., Tubaro, S., 2005. Clustering of human actions using invariant body shape descriptor and dynamic time warping. In: IEEE Conference on Advanced Video and Signal Based Surveillance.
- Rodriguez, M.D., Ahmed, J., Shah, M., 2008. Action MACH: a spatio-temporal maximum average correlation height filter for action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Schuldt, C., Laptev, I., Caputo, B., 2004. Recognizing human actions: a local SVM approach. In: IEEE International Conference on Pattern Recognition.
- Shechtman, E., Irani, M., 2005. Space-time behavior based correlation. In: International Conference on Computer Vision and Pattern Recognition.
- Smith, S.M., Brady, J.M., 1995. ASSET-2: real-time motion segmentation and shape tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (8), 814–820.
- Sundaresan, A., Chowdhury, A.R., Chellappa, R., 2003. A Hidden Markov model based framework for recognition of humans from gait sequences. In: IEEE International Conference on Image Processing.
- Tao, D. et al., 2007. General tensor discriminant analysis and gabor features for gait recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (10), 1700–1715.
- Turaga, P., Chellappa, R., Subrahmanian, V.S., Udrea, O., 2008. Machine recognition of human activities: a survey. *IEEE Trans. Circ. Syst. Video Technol.* 18.
- Turk, M., Pentland, A., 1991. Eigenfaces for recognition. *J. Cognitive Neurosci.* 3 (1).
- Vail, D., Veloso, M., Lafferty, J., 2007. Feature selection in conditional random fields for activity recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems.
- Weinland, D., Boyer, E., Ronfard, R., 2007. Action recognition from arbitrary views using 3D exemplars. In: International Conference on Computer Vision.
- Wu, Y., Yu, T., 2006. A field model for human detection and tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (5), 753–765.
- Yamato, J., Ohya, J., Ishii, K., 1992. Recognizing human action in time sequential images using Hidden Markov model. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Yan, P., Khan, S.M., Shah, M., 2008. Learning 4D action feature models for arbitrary view action recognition, In: IEEE Conference on Computer Vision and Pattern Recognition.
- Yang, Y., Liu, J., Shah, M., 2009. Video scene understanding using multi-scale analysis. In: International Conference on Computer Vision.
- Yuan, Y., Zheng, H., Li, Z., Zhang, D., 2010. Video action recognition with spatio-temporal graph embedding and spline modeling. In: IEEE International Conference on Acoustics, Speech and Signal Processing.
- Zelnik-Manor, L., Irani, M., 2001. Event-based analysis of video. In: IEEE Conference on Computer Vision and Pattern Recognition.
- Zheng, H., Li, Z., Fu, Y., 2009. Human action recognition with luminance field trajectory projection and alignment. In: IEEE International Conference on Multimedia and Expo.

This page is intentionally left blank

Soft Biometrics for Surveillance: An Overview

*D.A. Reid¹, S. Samangooei¹, C. Chen², M.S. Nixon¹,
and A. Ross²*

¹*School of Electronics and Computer Science,
University of Southampton, UK*

²*Lane Department of Computer Science and Electrical Engineering,
West Virginia University, USA*

Abstract

Biometrics is the science of automatically recognizing people based on physical or behavioral characteristics such as face, fingerprint, iris, hand, voice, gait, and signature. More recently, the use of soft biometric traits has been proposed to improve the performance of traditional biometric systems and allow identification based on human descriptions. Soft biometric traits include characteristics such as height, weight, body geometry, scars, marks, and tattoos (SMT), gender, etc. These traits offer several advantages over traditional biometric techniques. Soft biometric traits can be typically described using human understandable labels and measurements, allowing for retrieval and recognition solely based on verbal descriptions. Unlike many primary biometric traits, soft biometrics can be obtained at a distance without subject cooperation and from low quality video footage, making them ideal for use in surveillance applications. This chapter will introduce the current state of the art in the emerging field of soft biometrics.

Keywords: soft biometrics, surveillance, survey, fusion, identification

1. Introduction

Biometrics provides an automated method to identify people based on their physical or behavioral characteristics. Classical examples of biometric traits include fingerprints, irises, and faces which have been evaluated and demonstrated to be useful in several different applications ranging from laptop access to border control systems. Although these traits have been successfully incorporated in operational



Fig. 1. Surveillance frame displaying a few challenges in establishing identity of individuals in surveillance videos.¹

systems, there are several challenges that are yet to be addressed. For example, the utility of these traits significantly decreases when the input data is degraded or when the distance between the sensor and the subject increases. Thus, the use of alternate human attributes may be necessary to establish an individual's identity.

Soft biometric traits are physical or behavioral features which can be described by humans. Height, weight, hair color, and ethnicity are common examples of soft traits: they are not unique to the individual but can be aggregated to provide discriminative biometric signatures. Although these types of biometric traits have only been recently considered in biometrics, they have tremendous potential for human identification by enhancing the recognition performance of primary biometric traits.

Identification from a distance has become important due to the ever-increasing surveillance infrastructure that is being deployed in society. Primary biometric traits capable of identifying humans from a distance, viz., face and gait, are negatively impacted by the limited frame rate and low image resolution of most CCTV cameras. Figure 1 shows an example of a typical CCTV video frame. This frame shows a suspect in the murder of a Hamas commander in Dubai in 2010. The murder involved an 11-strong hit squad using fake European passports to enter the country. All the members of the hit squad used disguises including wigs and fake beards during the operation. From Fig. 1 it can be observed that although the image is at low resolution and the subjects' face and ocular features are occluded, a number of soft biometric features such as hair color, skin color, and body geometry can be deduced. Soft biometric traits can be extracted from very low quality data such as those generated by surveillance cameras. They also require limited cooperation from the subject and can be non-intrusively obtained, making them ideal in surveillance applications.

One of the main advantages of soft biometric traits is their relationship with conventional human descriptions (Samangooei et al., 2008); humans naturally use soft biometric traits to identify and describe each other. On the other hand,

¹ Arabian Business <http://www.arabianbusiness.com/for-hamas-murder-suspects-40450.html>.

translating conventional primary biometric features into human descriptive forms may not always be possible. This is the semantic gap that exists between how machines and people recognize humans. Soft biometrics bridge this gap, allowing conversion between human descriptions and biometrics. Very often, in eyewitness reports, a physical description of a suspect may be available (e.g., “The perpetrator was a short male with brown hair”). An appropriate automation scheme can convert this description into a soft biometric feature set. Thus, by using soft biometrics, surveillance footage archives can be automatically searched based on a human description.

Biometric traits should exhibit limited variations across multiple observations of a subject and large variations across multiple subjects. The extent of these variations defines the discriminative ability of the trait and, hence, its identification potential. Soft biometrics, by definition, exhibit low variance across subjects and as such rely on statistical analysis to identify suitable combinations of traits and their application potential. This chapter examines the current state of the art in the emerging field of soft biometrics. Section 2 introduces the performance metrics used in biometrics. Section 3 discusses how soft traits can be used to improve the performance of classical biometric systems based on primary biometric traits. Using soft biometric traits to identify humans is reviewed in Section 4. Identifying gender from facial images is explored in Section 5. Finally, Section 6 explores some of the possible applications of soft biometrics.

2. Performance metrics

The variation between multiple observations of an individual (intra-class variance) and the variation between subjects (inter-class variance) define the performance of a biometric system. If the intra-class variance of a biometric trait is low, then the trait is said to demonstrate permanence and repeatability. If the inter-class variance is high, then that biometric trait can be successfully used to distinguish between people.

Once an unknown subject’s biometric signature (i.e., the feature set extracted from the biometric data) has been determined, the system must then identify the subject based on a database containing a labeled set of biometric signatures. The labels correspond to the identity of the subjects in the database. This is typically performed by calculating the similarity between the input biometric signature (known as the probe) and the signatures within the database (known as the gallery). Generally, each identity in the database will be ranked based on this similarity measure, producing an ordered list of identities. The rank-1 retrieval performance of the biometric system details the probability of the correct identity being first in this ordered list.

While the above definition is for an *identification* system, it is possible for a biometric system to *verify* an individual’s identity. In such a scenario, the input biometric signature is labeled with an identity. Thus, the input signature is only compared against those biometric signatures in the database having the same identity label. If the similarity measure exceeds a threshold, then a “match” or an “accept” is said to have occurred; else, it is deemed to be a “non-match” or a “reject.”

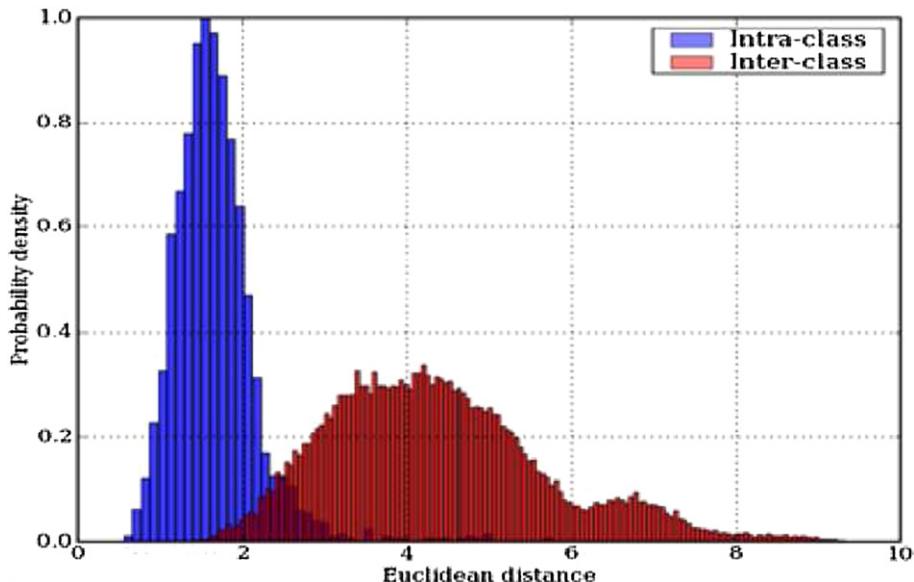


Fig. 2. Example intra/inter-class distributions.

When attempting to verify an identity, two errors can occur. A false accept (FA) occurs when the input signature is incorrectly “matched” against a different identity in the database. A false reject (FR) occurs when the input signature is incorrectly “rejected” when matched against the true identity in the database. These errors occur when (a) the biometric signatures of two subjects are very similar or (b) the biometric signature of a single subject varies with time. Typically, a threshold is chosen to define the similarity required for a “match.” This threshold determines the number of FA and FR errors. Figure 2 shows the intra-class and inter-class distances (i.e., a dissimilarity measure) of a sample biometric recognition algorithm. To achieve the best performance, the chosen threshold must minimize the FA and FR errors by separating the intra-class and inter-class distributions as optimally as possible. Receiver Operating Characteristic (ROC) curves show the trade-off between FA and FR errors at different threshold values; an example is shown in Fig. 3. The accuracy of two different ROC curves can be easily assessed by locating the point where the FA rate equals the FR rate: this is known as the equal error rate (EER). The EER provides a method of assessing the recognition performance of different biometric systems.

3. Incorporating soft biometrics in a fusion framework

Primary biometric traits such as face, fingerprints, and iris can suffer from noisy sensor data, non-universality, and lack of distinctiveness. Further, in certain applications, these traits may fail to achieve high recognition rates. Multimodal biometric systems (Ross et al., 2006) can solve these problems by combining multiple biometric traits, resulting in a biometric signature that is robust and more distinctive.

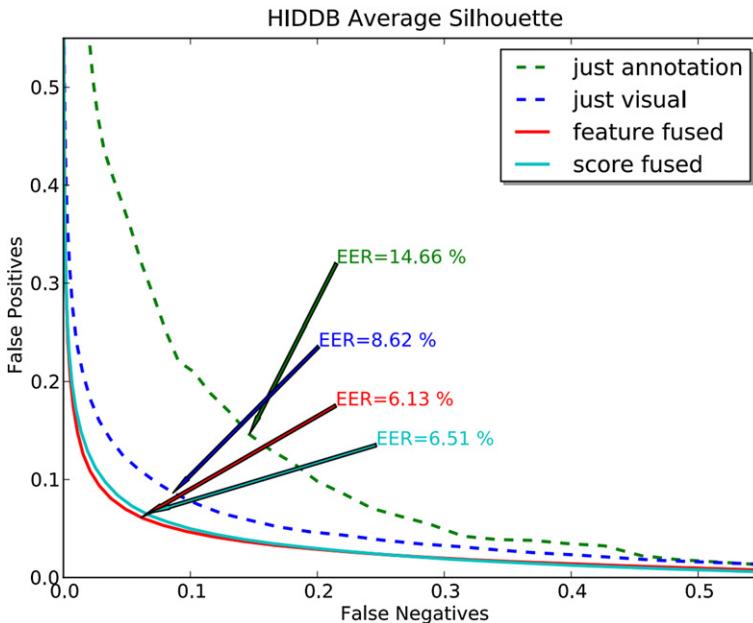


Fig. 3. Examples of ROC curves of four different techniques (Samangooei, 2010).

Multimodal systems offer improved performance, but the time taken to verify users can drastically increase thereby causing inconvenience to the subjects and reducing the throughput of the system. Soft biometric traits have been investigated to solve this problem (Jain et al., 2004b).

Jain et al. (2004a,b,c) experimented with the integration of soft biometrics in a biometric system. The primary biometric system compares the input biometric signature obtained from a user against each subject in the database. This determines the probability, $P(\omega_i|\mathbf{x}), i = 1, 2, \dots, n$, where n is the number of subjects within the database and $P(\omega_i|\mathbf{x})$ is the probability that the identity of the input primary feature vector \mathbf{x} is subject ω_i . The secondary soft biometric system uses one or more soft traits to confirm the output of the primary biometric system. The authors used height, gender, and ethnicity for this purpose. Gender and ethnicity were automatically obtained from facial images using the technique discussed in (Lu and Jain, 2004). The height data was not available within the test data and, hence, a random height was assigned to each user. The soft biometric feature vector \mathbf{y} updates $P(\omega_i|\mathbf{x})$ resulting in $P(\omega_i|\mathbf{x}, \mathbf{y})$ that is calculated using Bayes' theorem:

$$P(\omega_i|\mathbf{x}, \mathbf{y}) = \frac{P(\mathbf{y}|\omega_i)P(\omega_i|\mathbf{x})}{\sum_{i=1}^n P(\mathbf{y}|\omega_i)P(\omega_i|\mathbf{x})}. \quad (1)$$

Experiments were performed on a 263-subject database using both multimodal and unimodal primary biometric systems. The authors first considered the fusion of a fingerprint-based unimodal biometric system with a single soft biometric trait (one of height, gender, and ethnicity). It was observed that fusion resulted

in improved accuracy compared to the fingerprint system. Height was seen to be more discriminative compared to gender and ethnicity, leading to a 2.5% increase in rank-1 retrieval accuracy—although this could be a result of the random generation of heights. Fusing all three soft biometric traits with fingerprints resulted in a 5% increase in rank-1 accuracy compared to using fingerprints alone. Finally, soft biometrics were used to improve a multimodal system featuring face and fingerprints. An improvement in rank-1 accuracy of 8% (over individual modalities) was observed when combining gender, height, and ethnicity information.

Jain et al.'s system showed the advantages of using soft biometric fusion in the context of both unimodal and multimodal biometric systems. Increasing the number of soft and primary biometric traits increases the uniqueness of a user's signature, leading to better discrimination between subjects. Alisto et al. (2006) obtained similar success using body weight and fat measurements to improve fingerprint recognition, reducing the total error rate of 62 test subjects from 3.9% to 1.5%.

One concern, however, is the need for an automated technique to weight the soft biometric traits (Jain et al., 2004a). Marcialis et al. (2009) observed that certain soft biometric traits are only useful for a limited set of users. Their work only used soft biometric fusion when the user exhibited an uncommon soft trait thereby bypassing difficulties involved in weighting individual traits. It was assumed that the uncommon soft biometric feature could help in identifying a user from a set of possible candidate identities retrieved using primary biometric traits. An experiment fusing face with ethnicity and hair color was developed to verify this assumption. When fusing face and hair color (when uncommon), the EER on a database of 100 subjects fell from 6% to 4.5%. This paper clearly detailed the importance of uncommon traits and their ability to identify people. However, the use of hair color limits this technique to small databases and opens itself to spoof attacks.

The idea of utilizing uncommon traits was extended by Park and Jain (2010) and Jain and Park (2009) to identify people using facial marks. These marks include features such as scars, moles, freckles, acne, and wrinkles. Although not always permanent, most facial marks appear to be temporally invariant. The system proposed by the authors utilized facial marks, ethnicity, and gender to improve unimodal face recognition. One of the major advantages of facial marks is their utility (compared to automated facial matching) in courts of law since they are more descriptive and human understandable. In Park and Jain (2010), marks were characterized as salient localized regions on the face. Blob detectors based on the Laplacian of Gaussian are ideal for detecting such regions.

With respect to images, the second-order Laplace differential operator highlights changes in intensity. This is useful for detecting edges—which characterize the boundary between two regions of different intensity. The image is typically pre-smoothed to counter the operator's sensitivity to noise. Localized regions are identified as areas of either strong positive or negative responses for dark and light marks, respectively. Identified regions are highly dependent on the size of the mark and the size of the Gaussian kernel used for smoothing. To detect blobs of different sizes, the Laplacian operator is convolved at multiple scales by changing the size of the Gaussian kernel. In Park and Jain (2010), the resulting marks (blobs) were encoded into a 50-bin histogram and matched using the histogram intersection method. A commercial facial recognition system's EER was reduced from 3.85%

to 3.83% using facial marks. While this is a small reduction in EER, it demonstrates the utility of this soft biometric trait. Facial marks are especially beneficial when dealing with occluded or off-frontal face images. In their work, the authors artificially generated several examples of occluded face images, all of which were not recognized by the commercial facial recognition system. Upon using facial marks, the identities of subjects were correctly retrieved on average at rank 6. This demonstrates the benefit of utilizing uncommon traits and marks for human recognition in operational scenarios.

Thus, soft biometric fusion, when appropriately designed, can improve the accuracy of primary biometric systems with minimal inconvenience to the user. Soft traits can be used to either confirm results obtained from a classical biometric system or reduce the search space by filtering large databases. Soft biometric fusion is well suited for incorporation in security applications where speed and convenience are important. Further, in forensic applications, soft biometrics may help in confirming the identity of a subject.

4. Human identification using soft biometrics

As stated in the introduction, soft biometric traits lack the distinctiveness and permanence to accurately identify a person. This definition remains true when dealing with single traits, but has been shown to be partially overcome when dealing with multiple soft biometric traits. Dantcheva et al. (2010) likens this to obtaining a single ridge of a fingerprint or a small section of the iris: these would not be unique enough to identify a subject. However, by agglomerating many such features a reasonably unique signature can be constructed. Soft traits have some advantages compared to classical unimodal and multimodal systems.

One advantage of soft biometric systems is the bridging of the semantic gap between biometric traits and human descriptions. Soft biometric traits use human understandable descriptions (for example, height, hair color, and gender) and as a result can be naturally understood. This also negates the requirement of obtaining biometric data of subjects before identification, allowing previously unencountered subjects to be identified using human descriptions. This presents exciting possibilities such as searching surveillance footage and databases based solely on an eyewitness's description.

The two most popular traits for identification-at-a-distance are face (Zhao et al., 2003) and gait (Nixon and Carter, 2006). These can suffer from the poor sensor quality of most CCTV cameras. Low resolution can seriously impair facial recognition, and low frame rates (sometimes even time-lapse cameras) obscure the motion of the human body required for gait recognition. In contrast, several soft traits can often be obtained from very poor quality video or images. This has huge potential for immediate real-world use without upgrading the vast surveillance infrastructure.

Ailisto et al. (2004) presented a soft biometric system aimed at addressing concerns of privacy, identity theft, and the obtrusive nature of previous biometric solutions. Their system used unobtrusive and privacy preserving soft traits, including height, weight, and body fat percentage. The system had applications in

low-risk convenience scenarios where a relatively small number of people required identification, such as homes, small offices, and health clubs. Height, weight, and body fat were obtained from 62 subjects to mimic the target application environment. A monotonically decreasing penalty function was used to score the difference between stored measurements (templates) and the observed measurement (probe). A normal cumulative distribution was used to compute the scores:

$$Score = 2 \times (1 - P(Z \geq (|w_m - w_o|/c))). \quad (2)$$

Here, w_m is the model measurement, w_o is the observed measurement, c is a scaling factor based on an estimation of the variance of w_m , and $P(\cdot)$ is the standard normal cumulative distribution. Single modalities were shown to be very weak, with weight being the most distinctive resulting in a 11.4% total error rate (total false accepts and rejects). A combination of weight and height resulted in a 2.4% total error rate and the rank-5 retrieval accuracy was 100%. Using just three easy-to-obtain soft features allowed a database of 62 subjects to be sufficiently differentiated for the target application.

[Samangooei and Nixon \(2010\)](#) developed a soft biometric system that focused on bridging the semantic gap by allowing conversions between a primary biometric trait and a set of 23 soft biometric traits (Table 1). This important step allowed subjects to be searched using either previously obtained biometric data or a human description of a previously unknown subject. Each soft biometric trait was described using multiple categorical labels, also referred to as semantic terms, allowing for easy human interpretation. A primary biometric signature was used to understand the subjective meaning behind each label. It is important to note that, in principle, any physical representation of the subject, (a) that is a standard representation and (b) that encompasses the person's soft attributes, can be used. The use of gait signatures fulfills both these requirements, and permits the identification of subjects using gait signatures. By learning the relationships between the visual gait signature and the soft biometric features, the technique can be used to automatically label people based on their physical characteristics—thus converting gait signatures to human descriptions automatically.

Latent semantic analysis (LSA) ([Deerwester et al., 1990](#)), that is extensively used in document analysis, was employed to learn the structure between the two representations. The technique creates a vector space model which discovers a latent semantic structure based on the occurrences of features within documents. In this case, the documents are the subjects and the features are (a) the pixels within the biometric signature and (b) the semantic terms. By determining the co-occurrence between the features, we can detect the relationship between gait features and the semantic labels based on probability of occurrence.

The vector space is constructed using a set of subjects with semantically annotated gait signatures. [Samangooei et al. \(2008\)](#) obtained soft biometric labels for each subject in the Soton gait database ([Shutler et al., 2002](#)). The feature vectors of the subjects in the training set were combined to create the term-document matrix \mathbf{O} . This occurrence matrix embodies the underlying structure between the gait signature and the soft biometric features. This structure is hidden under a majority of irrelevant occurrences between features. By removing the irrelevant relationships (noise) the

Table 1
Semantic traits and corresponding terms

Trait	Terms
Arm length	[Very Short, Short, Average, Long, Very Long]
Arm thickness	[Very Thin, Thin, Average, Thick, Very Thick]
Chest	[Very Slim, Slim, Average, Large, Very Large]
Figure	[Very Small, Small, Average, Large, Very Large]
Height	[Very Short, Short, Average, Tall, Very Tall]
Hips	[Very Narrow, Narrow, Average, Broad, Very Broad]
Leg length	[Very Short, Short, Average, Long, Very Long]
Leg shape	[Very Straight, Straight, Average, Bow, Very Bowed]
Leg thickness	[Very Thin, Thin, Average, Thick, Very Thick]
Muscle build	[Very Lean, Lean, Average, Muscly, Very Muscly]
Proportions	[Average, Unusual]
Shoulder shape	[Very Square, Square, Average, Rounded, Very Rounded]
Weight	[Very Thin, Thin, Average, Fat, Very Fat]
Age	[Infant, Pre-Adolescence, Adolescence, Young Adult, Adult, Middle aged, Senior]
Ethnicity	[Other, European, Middle Eastern, Far Eastern, Black, Mixed]
Sex	[Female, Male]
Skin color	[White, Tanned, Oriental, Black]
Facial hair color	[None, Black, Brown, Blond, Red, Grey]
Facial hair length	[None, Stubble, Moustache, Goatee, Full Beard]
Hair color	[Black, Brown, Blond, Grey, Red, Dyed]
Hair length	[None, Shaven, Short, Medium, Long]
Neck length	[Very Short, Short, Average, Long, Very Long]
Neck thickness	[Very Thin, Thin, Average, Thick, Very Thick]

underlying semantic structure can be observed and an appropriate vector space can be constructed. Noise is removed by finding a rank-reduced approximation of the co-occurrence matrix. Singular Value Decomposition (SVD) is utilized to factorize the matrix allowing a rank reduced version to be determined. First, the matrix **O** is factorized into three matrices such that:

$$\mathbf{O} = \mathbf{TSD}^T. \quad (3)$$

Here, **T** and **D** are orthogonal matrices and **S** is a diagonal matrix. **S** contains the singular values from **O** and the matrices **T** and **D** contain the left and right singular vectors of **O**. By reducing the rank of these matrices, the dimensionality of the problem is reduced, resulting in an approximation of **O**. This approximation will ideally retain the most important information within **O**; this has been explored and proven in certain situations by Papadimitriou et al. (2000). The reduced rank *k* determines how many dimensions the data is condensed to and ultimately how much

information is removed. The diagonal matrix \mathbf{S} consists of r diagonal values that are ordered by size (and the corresponding row and column permutations applied to \mathbf{D} and \mathbf{T}). By removing the smallest singular values, the majority of the information is retained.

Once the noise has been removed and the semantic structure has been discovered, a k dimensional semantic vector space can be constructed. This space will consist of both features and subjects. The distance between these entities will correspond to their similarity or relevance. After the semantic space has been created, it can be queried to derive information about the structure. Comparisons can be made by finding the distance between the entities within the semantic space. Automatic soft biometric trait annotation is accomplished by inserting a test subject's gait signature into the semantic space. The closest and, hence, the most relevant soft biometric terms are then assigned to the subject resulting in a semantically annotated gait signature. The results from this approach were modestly successful showing an accuracy of 68% when determining semantic labels automatically from gait signatures (Reid and Nixon, 2010).

An interesting statistical analysis of these soft biometric traits was presented in (Samangooei, 2010). Each trait used to describe a person should be meaningful and provide additional information which differentiates the person from others. This property can be tested by determining the trait's ability to significantly separate the subjects within the database. If the subjects are not separated by a trait, then it could be said that the trait lacks any discriminative power and is not beneficial to the description (for the given set of subjects). To assess the discriminative power of each trait individually, the author used one-way ANOVA (analysis of variance).

$$\text{F-ratio} = \frac{\text{total between-group variance}}{\text{total within-group variance}}, \quad (4)$$

$$= \frac{\sum_i n_i (\bar{X}_i - \bar{X})^2 / (K - 1)}{\sum_{ij} (X_{ij} - \bar{X}_i)^2 / (N - K)}. \quad (5)$$

Here, X_{ij} represents the j th observation of the soft biometric of the i^{th} user and n_i denotes the number of observations of the i^{th} subject. \bar{X}_i is the mean of the i^{th} user's observations and \bar{X} is the mean across all subjects' observations. K represents the number of subjects while N represents the number of traits. Table 2 shows each trait's F-ratio, where a higher F-ratio indicates traits which are more successful at separating individuals.

It can be observed that "global" traits like gender, ethnicity, and skin color have more discriminative power than physical traits, like leg thickness. This is most likely due to the difficulty of labeling continuous physical traits compared to the categorical nature of the global traits. Traits like shoulder shape, proportions, and leg shape have been shown to be less discriminative thereby revealing their inability to distinguish between users. This important statistical analysis identifies the significance of each trait within a description and can be used to remove traits that do not contribute to additional information.

A statistical analysis of soft biometric systems utilizing categorical descriptions of physical traits was performed in Dantcheva et al. (2010, 2011) to determine the reliability of such a system in larger operational settings. When using categorical

Table 2

F-ratio of several soft biometric traits based on the Soton gait database

Trait	F-ratio	Trait	F-ratio
Sex	383.70	Neck thickness	14.73
Skin color	149.44	Arm thickness	13.90
Ethnicity	96.10	Leg length	13.68
Hair length	79.05	Muscle build	12.85
Age	57.02	Leg thickness	11.61
Hair color	52.18	Hips	10.55
Facial hair length	25.72	Arm length	5.74
Height	25.14	Facial hair colour	5.61
Weight	20.75	Leg shape	3.25
Figure	20.69	Proportions	2.77
Chest	18.32	Shoulder shape	2.54
Neck length	15.57		

labels, it is important to consider the likelihood of a subject being indistinguishable from other subjects in the database: this is referred to as inter-subject interference (Dantcheva et al., 2010). Obviously the interference has a huge impact on the soft biometric system's performance and the number of traits recorded directly affects the probability of interference between subjects. The system developed within the project identified nine semantic traits, mainly focusing on facial soft biometrics. These include: the presence of a beard, mustache, and glasses, each containing two terms; the color of the skin, eye, and hair composed of three, six, and eight terms, respectively; body mass index consisting of four terms defined by population norms. Further, the color of clothing on the torso and legs were determined, each being labeled based on a set of eleven terms.

To understand the likelihood of interference, first the number of distinct combinations of the traits, p , must be calculated

$$p = \prod_{i=1}^{\lambda} \mu_i. \quad (6)$$

Here, λ is the number of soft biometric traits and μ is the number of terms associated with the corresponding trait. Based on the aforementioned nine traits $p = 557,568$. It must be noted that many of these combinations will not be realized when dealing with small databases, leaving a set of non-empty combinations F ranging from 0 to $\min(p, N)$ where N is the number of people in the database. The authors assumed that the distribution of subjects across the p possible combinations is uniform and that $N < p$. Let C_F be the total number of feature vectors which result in the set of F combinations, where F ranges between 1 and N . Then

$$C_F = \frac{p!}{(p - F)!} \frac{N!}{(N - F)!} F^{N-F}. \quad (7)$$

Here, the first term describes the total number of ways F combinations can be chosen to host the subjects. The second term describes the possible ways the F combinations can be filled with F subjects. The last term describes the number of ways the F combinations can be associated with the remaining $N - F$ subjects. Giving the distribution of $F, P(F)$, as

$$P(F) = \frac{C_F}{\sum_{i=1}^N C_i}, \quad (8)$$

where the mean of $F, E[F] = \sum_{F=1}^N FP(F)$. Finally the probability of error averaged over all subjects results as:

$$P(\text{err}) = 1 - \frac{E[F]}{N}. \quad (9)$$

Figure 4 shows the likelihood of interference occurring with N subjects where N ranges from 0 to 1000 subjects. The figure shows the probability of interference, $p(N)$, within a database of subjects and the probability of a randomly chosen subject from the database interfering with another subject (s), $q(N)$. Figure 4 clearly shows that with only 49 people a 50% chance of interference exists. This likelihood of interference can be reduced by increasing the uniqueness of each subject's trait signature. Increasing the amount of possible combinations of terms is one possible method for achieving this—only if the new term combinations further discriminate between the subjects. This can be achieved by either increasing the number of traits or the detection of more terms per trait. In comparison, Samangooei et al. (2008)'s soft biometric system, featuring 23 traits, has 3.7×10^{15} possible combinations of semantic terms—decreasing the likelihood of interference. This important work clearly identified the need for optimizing the number of term combinations and its effects on interference and ultimately improving the performance of the soft biometric system. Further statistical studies are required to identify the optimal number of term combinations for target application environments, taking into account the expected distributions across different soft traits.

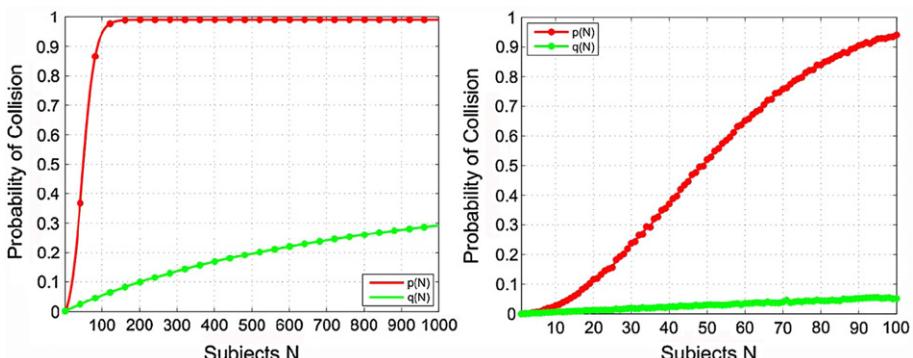


Fig. 4. Interference probability in a N -sized population ranging from 0 to 1000, and a magnified version showing 0–100 (Dantcheva et al., 2010).

4.1. Imputation

Human physical traits and appearance inherently contain structure, features frequently co-occur or have fixed relationships with other features. This occurs either due to social aspects (long hair is common in females), genetics (black hair is common within people of Asian descent), or the morphology of the human body (taller people are more likely to have longer legs). Imputation techniques are a statistical approach used to predict missing variables. Using such techniques missing soft biometric features can be predicted utilizing the structure within human appearance. This structure offers a basis to improve the recognition of soft biometric traits and to make soft biometric systems more robust to missing traits or occluded visual features.

Reid and Nixon (2010) extended the soft biometric system presented in Samangooei and Nixon (2010) and identified several scenarios where imputation would be critical. Occlusion is a major concern when using soft biometric systems in unconstrained environments. Visual features can be concealed by scenery, the person's own body (self occlusion) or covariates such as bags, hats, and clothing. Structure can be utilized to predict the soft biometric terms normally obtained from occluded visual data, to provide a more complete description of the person. Similarly human descriptions obtained from witnesses, which could be used to identify people, are often unreliable and incomplete. Utilizing structure, these erroneous or missing

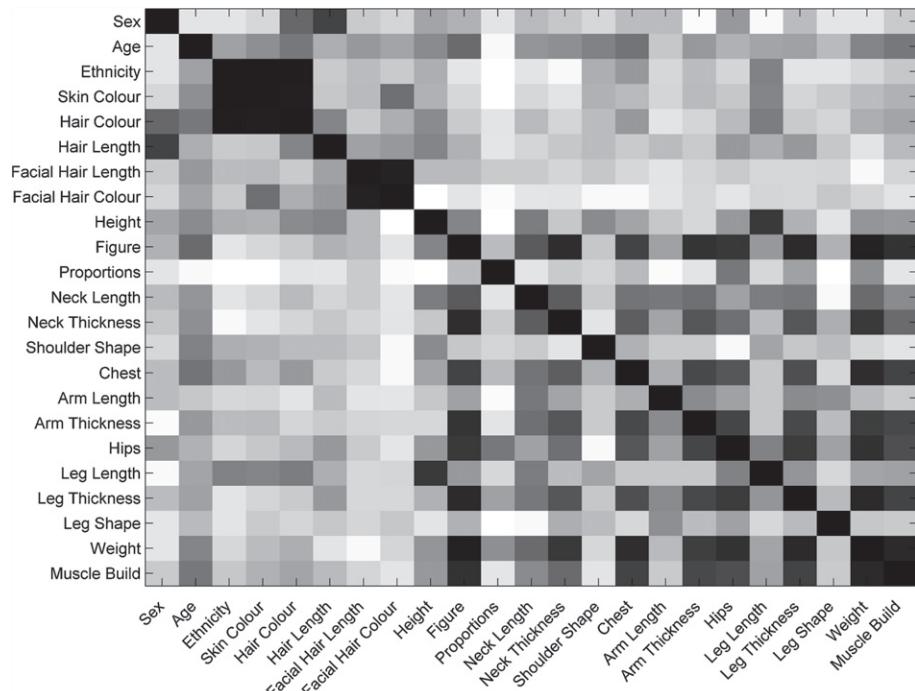


Fig. 5. Correlation (Pearson's r) between soft biometric features.

labels can be assessed and the most probable labels based on previous experience can be used to refine the description.

To verify that structure is present within human descriptions a correlation matrix was produced. This shows correlation between soft biometric traits based upon their occurrences within the Soton gait database (Shutler et al., 2002). Figure 5 shows the correlation matrix where darker cells represent more correlated features. It can be observed that large amounts of correlation occur within the 23 traits. The most prominent is that between skin color, hair color, and ethnicity, which can be seen in the top left corner. This relationship details the genetic likelihood that people from certain ethnic backgrounds are likely to have a certain skin color and hair color. Another interesting region within the figure is the lower right corner which details the relationship between physical attributes like weight, arm length, and leg thickness. Strong relationships exist between weight and thickness as well as height and length, producing high correlation. Some areas of the matrix contain little correlation, for example, the relation between weight and facial hair. The low correlation does not mean that there is no relationship between the features, only that it is not prevalent within the dataset.

It was first decided to utilize this known correlation to predict missing data. If the missing semantic trait is highly correlated with another trait, then it is beneficial to exploit this relationship to predict the missing term. The technique uses a method similar to the k nearest neighbor (k NN) classification technique. Each subject within the training set is compared to the subject containing the missing value. Typically this comparison involves finding the Euclidean distance between the two subjects' feature vectors. This has been modified to make use of the known correlation between traits. The similarity of a neighbor's trait is weighted by the correlation between that trait and the subject's missing trait. This favors neighbors with the same labels for those traits that have a strong relationship with the missing trait. The similarity between two subjects is determined as shown in Eq. (10) where X is the neighbor, Y is the subject with missing trait i . N is the total number of semantic traits and the matrix \mathbf{C} contains the correlation between two semantic traits (values range from $[-1,1]$). The missing trait is predicted by taking the mode of the corresponding trait within the k nearest neighbors.

$$\text{Similarity}(X) = \frac{1}{N} \sum_{j=1}^N |\mathbf{C}_{i,j}| (1 - |X_j - Y_j|). \quad (10)$$

Correlation cannot determine relationships between terms and traits, though it is adequate for determining linear relationships between traits. Table 3 shows the observations of skin color and hair color obtained from the Soton gait database. It can be observed that some terms, for example white skin, show more variance when compared to other terms from the same trait, for example oriental skin. By determining the correlation over all terms within a trait, potentially strong ties between terms, for example oriental skin and black hair, are being lost. By observing a term's ability to predict a missing trait better accuracy can be achieved. It can be seen that ideal terms to predict hair color contain the least variance over their occurrences with hair color. This important property can be used to estimate the ability of a term to predict a missing trait and can be used to weight the similarity

Table 3

Percentage of observations of hair colour and skin color

	Black	Blond	Brown	Grey	Red	Dyed
Black	1	0	0	0	0	0
Oriental	1	0	0	0	0	0
Tanned	0.86	0	0.14	0	0	0
White	0.01	0.22	0.7	0.03	0.01	0.03

when looking for the k nearest neighbors. Calculating the entropy of all the elements within a row provides a measure of certainty. This shows how successful the term is at predicting the missing trait. The entropy is used to weight neighbor's similarity. The similarity between two subjects is determined as shown in Eq. (12) where X is the neighbor, Y is the subject with missing trait i , which is composed of T_i terms. The matrix \mathbf{P} contains the percentages of observations (Table 3) between terms, such that $\mathbf{P}_{l,j}$ details the observations of term j with term l . N is the total amount of semantic terms within the semantic feature vector.

$$H(j) = 1 - \sum_{l=1}^{T_i} -\mathbf{P}_{j,l} \log \mathbf{P}_{j,l}, \quad (11)$$

$$\text{Similarity}(X) = \frac{1}{N} \sum_{j=1}^N H(j)(1 - |X_j - Y_j|). \quad (12)$$

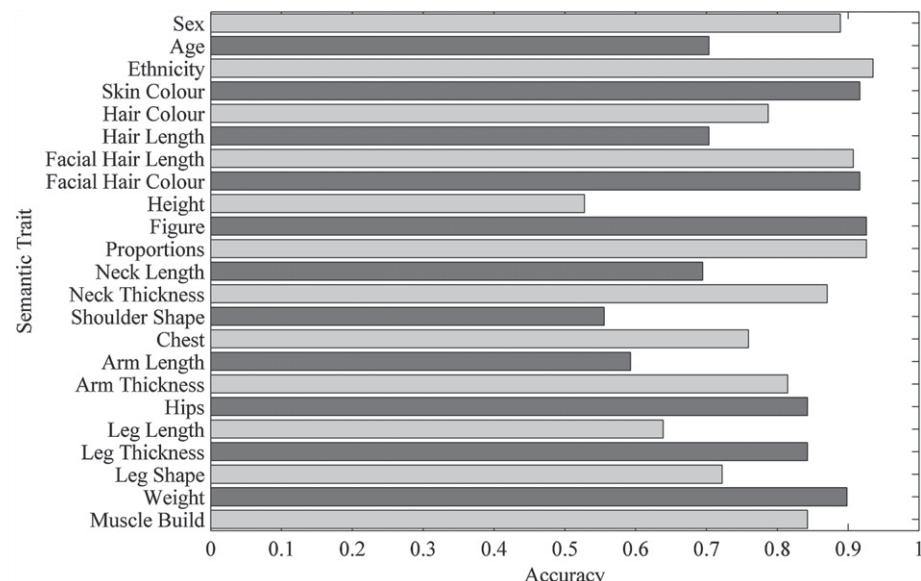
Fig. 6. Rebuilding soft data using entropy-based weighted k NN.

Figure 6 shows the accuracy of rebuilding each semantic trait using the entropy-based approach. It can be observed that the most successful traits are skin color and ethnicity, this is likely due to their strong correlation with other traits allowing accurate predictions of missing data.

Adjeroh et al. (2010) have also studied correlation and imputation in human appearance analysis. The most important difference with the previous study is the use of continuous data focusing on measurements of the human body. Data was gathered from the CAESAR anthropometric dataset which comprised of 45 human measurements or attributes for 2369 subjects. Like Reid and Nixon (2010), the relationships between the human measurements were first assessed using the Pearson correlation coefficient. To visualize the correlation, a correlation graph was created—shown in Fig. 7. This graph shows connections between traits if the correlation was stronger than a threshold value. This clearly confirms the structure within human appearance and highlights clusters of traits with strong correlation. It can be observed that the measurements generally fall into two groups, both of which have physical meaning: the 2D group which contains circumferences of body parts and the 1D group containing lengths and heights. These clusters suggest that only a few measurements would have to be known to predict the majority of the other traits.

The metrology predictability network was developed to predict missing traits based on the most suitable subset of observed traits. Like the previous paper it

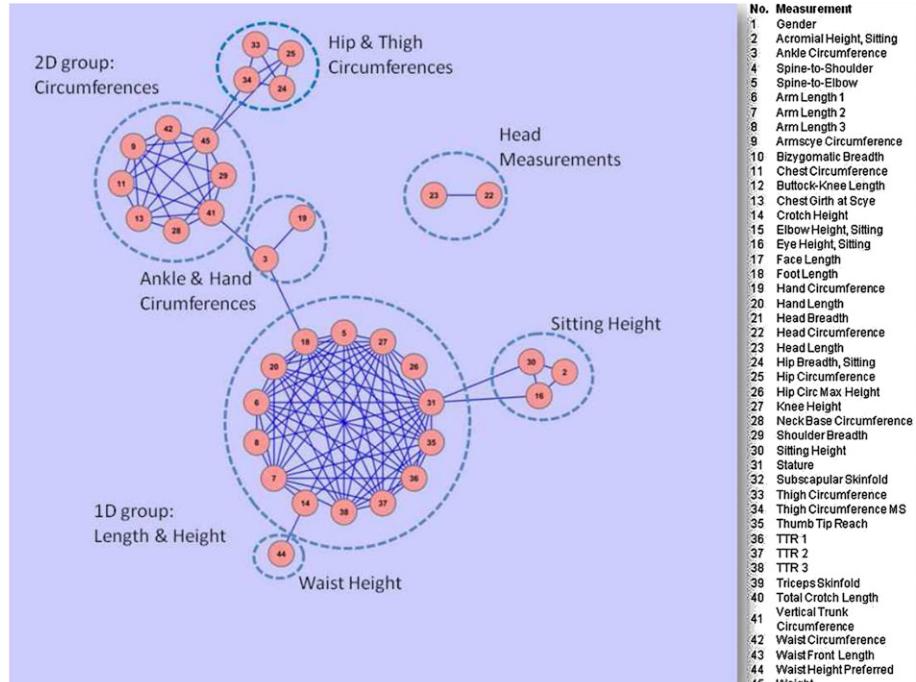


Fig. 7. The relations between measurements based on correlation (>0.81) (Adjeroh et al., 2010).

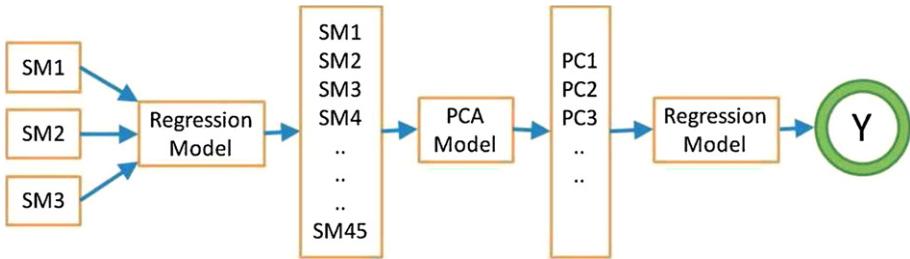


Fig. 8. Two-step prediction, using three seed measurements (Adjerooh et al., 2010).

appeared intuitive to use the most correlated traits to make the prediction. Using the correlation graph any nodes linked to the missing node are used in the prediction process. Traits which have been shown to accurately predict the missing trait are also considered. The expected error is assessed using multiple linear regression on training data from the CAESAR dataset

$$y = \mathbf{X}\beta + \epsilon, \quad (13)$$

where y is the response variable, \mathbf{X} is the input (predictor) variable matrix, β is the parameter vector, and ϵ is the error term. β is estimated using least squares such that:

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T y. \quad (14)$$

Thirty one prediction models were constructed, each varying the order of the model, the number of variables, and the variable combinations. Using a training set of measurements, the error predicting a trait using a single prediction model is assessed. These errors are used to create a predictability graph (similar to Fig. 7) denoting the ability of a trait to predict another trait accurately, where edges denote errors which are below a threshold.

The measurements obtained from the subject, called the seed measurements, are used to predict the missing traits. Some traits will be easy to predict due to their strong relationship with the seeds. These highly correlated traits are also used to predict traits with a weak relationship with the seeds. Principal component regression is used to predict the missing traits. An initial prediction is made using regression on the seed measurements, principal component analysis is used to reduce the measurements needed and then regression is applied to these features (Fig. 8). Experiments were conducted on the CAESAR dataset and 23 subjects from the CMU motion capture database. Four seed measurements were used—arm length, knee height, shoulder breadth, and standing height. Based on these seeds the remaining traits were predicted with an average mean absolute error of 0.041. Another experiment predicted all 41 measurements from just three seeds and used these measurements to predict the gender, resulting in a 88.9% correct gender classification rate.

It has been shown that human appearance contains an inherent structure and just a few seed measurements are required to accurately predict the remaining features. This redundancy is vital when dealing with occlusion in visual data.

Structure is inherent within human appearance and is echoed in human descriptions and biometric representations. Imputation is crucial in operational settings where visual data is often occluded and human descriptions are often erroneous or incomplete. By utilizing structure within the soft traits, issues with view invariance and the subjective and unreliable nature of human descriptions can be addressed.

5. Predicting gender from face images

The problem of automated gender classification has attracted significant attention in many areas such as soft biometrics (Jain et al., 2004a; Mahalingam and Kambhamettu, 2011), demographic classification (Gutta et al., 1998; Yang and Ai, 2007), and human computer interaction. Gender classification is a fundamental task for human beings, as many social activities depend on the precise identification of gender. In the context of biometrics, gender is viewed as a soft biometric trait that can be used to index databases or enhance the recognition accuracy of primary traits such as face (Jain et al., 2004a; Mahalingam and Kambhamettu, 2011).

The study of automated gender classification from face images dates back to the early 1990s (Golomb et al., 1990) and has recently gained attention in the context of attribute-based face recognition (Kumar et al., 2009). The problem of gender classification from face images can be posed as a two-class problem in which the input face image is analyzed and assigned to one of the two classes: male or female. The problem of gender classification from face images ranges from the visible to infrared spectra (Ross and Chen, 2011; Chen and Ross, 2011) and from constrained (Moghaddam and Yang, 2002; Baluja and Rowley, 2007; Makinen and Raisamo, 2008) to unconstrained images (Prince and Aghajanian, 2009; Toews and Arbel, 2009; Shan, 2010; Hu et al., 2011).

Golomb et al. (1990) trained a back-propagation neural network (BPNN) to identify gender from human face images at a resolution of 30×30 pixels. An average classification rate of 91.9% on 90 exemplars was obtained compared to a human performance of 88.4%. Gutta et al. (1998) used hybrid classifiers consisting of an ensemble of radial basis function (RBF) networks and decision trees. The experiments were conducted on a collection of 3006 face images corresponding to 1009 subjects from the FERET database. The cross-validation results yielded an average accuracy of 96% for the gender classification task. Later on, Moghaddam and Yang (2002) utilized a support vector machine (SVM) for gender classification, based on low-resolution thumbnail face images of resolution 21×12 pixels. The average classification rate for fivefold cross-validation on 1755 FERET face images was 96.62% with the use of the Gaussian RBF kernel. Baluja and Rowley (2007) presented the use of an Adaboost classifier to identify the gender of a person from a low-resolution face image. The proposed system was extremely fast and yet comparable to the SVM-based classifier. They reported an accuracy over 93% on a dataset of 2409 FERET faces images with resolution 20×20 pixels.

Although the use of raw pixel values of face images in (Golomb et al., 1990; Gutta et al., 1998; Moghaddam and Yang, 2002; Baluja and Rowley, 2007) results in very good gender classification performance, it is a simple feature representation scheme that may not be robust enough in some complex scenarios involving pose and illumination changes. Therefore, other types of feature extraction methods have also been proposed. Yang and Ai (2007) used Local Binary Pattern (LBP) histogram features (Ahonen et al., 2006) for gender feature representation, and the real Adaboost algorithm to learn the best local features for classification. Experiments were performed to predict the age, gender, and ethnicity information from face images. Similar work was presented in Sun et al. (2006), where LBP features and Adaboost classifier were combined to achieve better performance.

Other local-based descriptors have also been adopted in the work of gender classification. For example, Guo et al. (2009) evaluated gender classification results based on LBP, histograms of oriented gradients (HOG), and Biologically Inspired Features (BIF) with SVM as the classifier. It was demonstrated that gender prediction was affected by age variations on a large database. Wang et al. (2010) proposed a novel gender recognition method in terms of the Scale Invariant Feature Transform (SIFT) descriptor and shape contexts. Again, Adaboost was used to select features from face images to form the strong classifier. Other approaches utilized gender-specific information, such as hair, to enhance gender prediction (Lian and Lu, 2008), or genetic algorithms to select features encoding gender information (Sun et al., 2002). Cao et al. (2011) proposed the use of facial metrology for gender prediction.

All the aforementioned work mainly focus on datasets that were collected under well-constrained environments. Recently, gender classification on unconstrained real-world face images has been attempted. Chen and Lin (2010) built a gender classification system on real-world face images where the decision was based on the region surrounding the detected face. Shan (2010) proposed to use the boosted LBP features to represent face images and applied SVM to determine the gender on the Labeled Faces in the Wild (LFW) dataset. An accuracy of 94.44% was obtained on a dataset of 7,443 face images. Gallagher and Chen (2009) used social context information in real-world group images to accomplish gender classification. They argued that the structure information within the group provides meaningful context for individuals. For example, men were more likely to stand in the corner of a photograph than women. Gao and Ai (2009) targeted face gender classification on consumer images in a multiethnic environment. To overcome the non-uniformity of pose, expression, and illumination changes, they proposed a robust Active Shape Model (ASM) to normalize the face texture. The consideration of ethnic factors can help improve gender classification accuracy in a multiethnic environment. Recently, Toews and Arbel (2009) extended gender classification to images involving arbitrary viewpoints and occlusions. A viewpoint-invariant appearance model was learned for the object class and a Bayesian classifier was trained to identify the model features that indicate gender. In the work of (Prince and Aghajanian, 2009), images with unconstrained pose, expression, and light conditions were considered for gender classification based on additive logistic models.

With advancements in sensing technologies, thermal and near-infrared images are beginning to be used for face-related applications. For example, face recognition

in near-infrared (NIR) (Li et al., 2007; Zhang et al., 2010) and thermal (THM) (Socolinsky and Selinger, 2002) spectra has been motivated by the need to determine human identity in nighttime environments (Bourlai et al., 2011). Furthermore, changes in ambient illumination have lesser impact on face images acquired in these spectra than the visible spectrum. Current gender classification systems discussed in the literature have been designed for and evaluated on face images acquired in the visible spectrum. Little attention has been given to automatic gender classification from faces in the thermal or near-infrared spectrum. Ross and Chen (2011) attempted gender prediction on near-infrared face images. They later extended the work to gender prediction in the thermal spectrum (Chen and Ross, 2011). Compared to humans, machine learning-based approaches achieved better results on thermal face datasets.

In principle, a gender classification method can be divided into two components: (a) a feature extractor that extracts features from the face and (b) a feature classifier that assigns the extracted features into one of two classes—male or female. Feature extraction methods include the use of raw pixel face images (Moghaddam and Yang, 2002; Baluja and Rowley, 2007), Principle Component Analysis (PCA) (Balci and Atalay, 2002), Linear Discriminant Analysis (LDA) (Bekios Calfa et al., 2011), Independent Component Analysis (ICA) (Jain and Huang, 2004), LBP (Yang and Ai, 2007; Shan, 2010; Sun et al., 2006; Chen and Ross, 2011), and metrology (Cao et al., 2011). Some feature selection algorithms (Khan et al., 2005) have also been used to select gender specific features. Most gender classifiers are based on Neural Network (Golomb et al., 1990; Khan et al., 2005), Adaboost (Baluja and Rowley, 2007; Yang and Ai, 2007; Sun et al., 2006; Shan, 2010), Gaussian Process (GP) classifier (Kim et al., 2006), and SVM (Moghaddam and Yang, 2002; Guo et al., 2009; Chen and Ross, 2011). A systematic overview of methods for gender classification from face images in the visible and thermal spectra can be found in Makinen and Raisamo (2008) and Chen and Ross (2011), respectively.

An overview of gender classification methods and their accuracies is summarized in Table 4. It gives a brief insight into the different algorithms used in the past (Chen, 2011).

6. Applications

6.1. Continuous authentication

Most existing computers only authenticate users at the beginning of a session, leaving the system open to imposters until the user logs out. Continuous user authentication provides a method to continually confirm the identity of the user. Conventional biometric modalities such as face and fingerprint are either inconvenient for continuous operation or difficult to capture when the user is not explicitly interacting with the sensor. Soft biometrics offers a potential solution to this problem (Niiuma et al., 2010) by using features like the color of the user's clothes and facial skin.

When the user is initially authenticated using facial recognition and a password, soft biometric traits are obtained and recorded. Throughout the session the user is authenticated using these traits, without enforcing a strict posture or requiring

Table 4

Overview of some studies on gender determination from a face image

Author & Year	Features	Classifier	Database, size	Accuracy (%)
Golomb et al. (1990)	Raw pixels	Neural network	Private, 90	91.9
Getta et al. (1998)	Raw pixels	Hybrid classifier	FERET, 3006	96.0
Moghaddam and Yang (2002)	Raw pixels	SVM	FERET, 1755	96.62
Jain and Huang (2004)	ICA	LDA	FERET, 500	99.3
Khan et al. (2005)	PCA	Neural network	Private, 400	88.7
Sun et al. (2006)	LBP	Adaboost	FERET, 2000	95.75
Kim et al. (2006)	Raw pixels	GPC	AR, 515	97.0
Baluja and Rowley (2007)	Raw pixels	Adaboost	FERET, 2409	93.0
Yang and Ai (2007)	LBP	Adaboost	Private, 3540	96.32
Xia et al. (2008)	LBP, Gabor	SVM	CAS-PEAL, 10784	93.74
Gao and Ai (2009)	ASM	Adaboost	Private, 1300	92.89
Toews and Arbel (2009)	SIFT	Bayesian	FERET, 994	83.7
Shan (2010)	LBP	Adaboost	LFW, 7,443	94.44
Guo et al. (2009)	LBP, HOG, BIF	SVM	YGA, 8,000	89.28
Wang et al. (2010)	SIFT, context	Adaboost	FERET, 2409	95.0
Bekios Calfa et al. (2011)	PCA	LDA	FERET, 994	93.33
Chen and Ross (2011)	LBP	SVM	CBSR NIR, 3200	93.59
Cao et al. (2011)	Metrology	SVM	MUCT, 276	86.83
Hu et al. (2011)	Filter banks	SVM	Flickr images, 26700	90.1

constant verification. Facial recognition is also used periodically, when the biometric data is available, to guard against spoof attacks. Histograms of the various colors are gathered and the Bhattacharyya coefficient (Bhattacharyya, 1943) is used to calculate the similarity of two histograms, by measuring the amount of overlap.

In one experiment, a database of 20 subjects was constructed. Each subject was asked to perform six actions including turning their heads, leaning back in their chair, stretching arms, and walking away from the computer. The average false rejection and false acceptance over all the recorded actions were 4.16% and 0%, respectively. Soft biometrics has been shown to provide secure continuous user authentication while being robust to the user's posture and not requiring manual registration of the soft biometric traits for each session.

6.2. Surveillance and re-identification

CCTV cameras have been widely introduced and accepted (Webster, 2009; Helten and Fischer, 2004). Their primary role within society is to assist in the fight against crime (Webster, 2009). This involves deterring and detecting crime, reducing the fear of crime, and to provide evidence when crime does occur. There has been considerable investment into the CCTV infrastructure (particularly in the UK) but currently these cameras (and the ensuing footage) are still generally monitored by humans. Due to the number of cameras within most cities, operators cannot monitor the data in intricate detail. This means looking for a single person can be time consuming and

prone to mistakes. Soft biometrics can potentially solve these problems by providing a method for searching surveillance footage using human descriptions.

Soft biometrics offer several benefits over other forms of identification-from-a-distance. Face recognition often requires good resolution images and gait recognition requires good frame rates. In comparison, certain soft biometric traits can be obtained from low-resolution and low-frame rate videos, and from an arbitrary viewpoint of the subject. Although not descriptive enough to identify people from large databases, they are adequate when dealing with smaller populations. The human compliant nature of soft biometric traits can also be exploited to allow searches based solely on a human description - possibly obtained from an eye witness. This allows for the use of soft biometrics when primary biometric identifiers cannot be obtained or when only a description of the person is available.

[Denman et al. \(2009\)](#) uses soft biometric traits to identify people using previous observations or human descriptions when traditional biometrics are unavailable. The height and color of the torso, legs, and head are used to model subjects. Identifying these three body components is done by first locating the person using background segmentation and then analyzing the color of moving pixels in each row. Large color differences are often found between the head, torso, and legs due to clothing that can be easily identified by examining color gradients. Average body proportions are used to identify the most likely color gradients representing the three desired regions. After the regions are located, a color histogram is recorded and the real-world height estimated. Heights are matched using average height and standard deviations, and color histograms are matched using the Bhattacharyya coefficient. The PETS 2006 surveillance database was used to test the system. This dataset features four cameras monitoring a train station: four recordings of 25 people were obtained. The system achieved an equal error rate of 6.1% when evaluated using the leave-one-out cross-validation scheme. These recordings included videos from two different viewpoints, demonstrating the view invariant nature of the selected soft traits. In comparison, primary biometric traits such as face, typically only work from one viewpoint.

Further work in soft biometrics has provided a technique to recognize subjects moving between multiple surveillance cameras in order to generate a rough framework for facial recognition ([Demirkus et al., 2010](#)). The technique uses gender, ethnicity, and session-based soft biometrics (skin color, upper and lower body clothing color, and hair color). Session-based soft biometrics are features which are reasonably constant for a short time period. These features, although not permanent, allow subjects to be identified when moving between different cameras. Once a person has been identified in the surveillance footage, the directional pose is determined. If the person is walking toward the camera, the face is analyzed to deduce ethnicity and gender, which is combined with the color-based traits that are extracted automatically. Gender and ethnicity are determined using support vector machines, and the color information is encoded using histograms. When a camera observes a new subject, their session-based features are compared to that of people previously observed by the camera network. If a match is found, the subject is given the same identity tag.

A custom low-resolution surveillance dataset was constructed featuring 100 subjects. An average correct classification rate of 60% and 83%, for gender and

ethnicity, respectively, was observed using just a resolution of 66×61 (pixels) facial images obtained from the video dataset. Gender and ethnicity were also used to partition the database of observed faces to speed up queries. The gender and ethnicity of the facial query were obtained and only faces featuring the same soft traits within the database were tested. The soft biometric partitioning reduced the time required for face recognition queries by almost a factor of 6 on a 600 subject database. Session-based soft biometrics are ideal for tracking people between cameras due to the speed in trait acquisition and their view invariant nature. Additional traits would allow for tracking in more crowded areas and would reduce the reliance on color, which is problematic if the cameras are not calibrated. Additional traits could also be used to partition the database further thereby reducing the time taken for primary biometric queries.

7. Conclusion

The use of soft biometric traits in automated human recognition systems has several benefits. It is, therefore, essential to carefully investigate issues related to its extraction and recognition capacity.

Surveillance footage is generally of inferior quality and so traditional forms of identification-at-a-distance cannot be easily used. Soft biometrics offers a solution in this regard but lacks the distinctiveness that is expected of biometric traits. Applications with smaller populations are ideal, such as tracking people within a camera network or identifying people who are known to be located within a certain area. View invariance is a key aspect when working with surveillance footage—a successful technique must identify soft biometric features from any view of the subject. Emphasis must be placed on finding practical ways of obtaining view invariant features (similar to Denman et al. (2009)) or developing methods to predict hidden features based on what can be observed.

Bridging the semantic gap between human descriptions and biometrics will allow us to use natural “verbal” descriptions to search databases and surveillance footage. This is not possible using primary biometric modalities and, hence, opens up new applications and exciting prospects. Currently, biometric systems require an example of the subject before identifying them; this means unrecorded subjects cannot be automatically found. Human descriptions provide a way of finding unknown subjects using the description alone, which is critical in many crime-scene investigations.

Human descriptions are often erroneous or incomplete due to the subjective nature of the descriptors and the psychological pressure experienced by eye witnesses. By utilizing the known pattern within human features and measurements, these descriptions could be corrected leading to a more informative and correct search criteria. The subjective nature of the descriptors could also be reduced by utilizing comparisons between people instead of absolute labels. Obtaining correct and detailed descriptions is vital when searching for suspects, and would allow soft biometric systems to function effectively.

Soft biometrics typically require less computation and data quality compared to other forms of identification-from-a-distance, making them cheap and

non-intrusive. The authors in Niinuma et al. (2010) clearly demonstrate the suitability of soft biometric traits for continuous user authentication. Determining applications which are suited to this form of biometric identification is essential for advancing the field. Statistical analysis underpins the performance of any biometric trait, and is vital for assessing the discriminability of feature measures as well as their recognition performance. By agglomerating multiple soft biometric features using fusion techniques, the recognition performance can be further enhanced leading to applicability in a wide variety of environments.

References

- Adjero, D., Cao, D., Piccirilli, M., Ross, A., 2010. Predictability and correlation in human metrology. In: Proceeding of IEEE International Workshop on Information Forensics and Security, pp. 1–6.
- Ahonen, T., Hadid, A., Pietikainen, M., 2006. Face description with local binary patterns: application to face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 28, 2037–2041.
- Ailisto, H., Lindholm, M., Makela, S.M., Vildjounaite, E., 2004. Unobtrusive user identification with light biometrics. in: Proceeding of NordiCHI, pp. 327–330.
- Ailisto, H., Vildjounaite, E., Lindholm, M., Makela, S., Peltola, J., 2006. Soft biometrics—combining body weight and fat measurements with fingerprint biometrics. *Pattern Recogn. Lett.* 27 (5), 325–334.
- Balci, K., Atalay, V., 2002. PCA for gender estimation: which eigenvectors contribute? In: International Conference on Pattern Recognition, pp. 363–366.
- Baluja, S., Rowley, H.A., 2007. Boosting sex identification performance. *Int. J. Comput. Vis.* 71 (1), 111–119.
- Bekios Calfa, J., Buenaposada, J., Baumela, L., 2011. Revisiting linear discriminant techniques in gender recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (4), 858–864.
- Bhattacharyya, A., 1943. On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math. Soc.* 35, 99–109.
- Bourlai, T., Kalka, N.D., Cao, D., Decann, B., Jafri, Z., Nicolo, F., Whitelam, C., Zuo, J., Adjero, D.A., Cukic, B., Jeremy, D., Larry, H., Ross, A., Schmid, N., 2011. Ascertaining human identity in night environments. In: Distributed Video Sensor Networks, pp. 471–478.
- Cao, D., Chen, C., Piccirilli, M., Adjero, D., Bourlai, T., Ross, A., 2011. Can facial metrology predict gender? In: International Joint Conference on Biometrics.
- Chen, C., 2011. Gender classification from facial images. West Virginia University, USA, Master's Thesis.
- Chen, D.-Y., Lin, K.-Y., 2010. Real-time gender recognition for uncontrolled environment of real-life images. In: International Conference on Computer Vision Theory and Applications, pp. 357–362.
- Chen, C., Ross, A., 2011. Evaluation of gender classification methods on thermal and near-infrared face images. In: International Joint Conference on Biometrics.
- Dantcheva, A., Dugelay, J., Elia, P., 2010. Soft biometrics systems: reliability and asymptotic bounds. In: BTAS, September, pp. 1–6.
- Dantcheva, A., Velardo, C., DAngelo, A., Dugelay, J., 2011. Bag of soft biometrics for person identification: new trends and challenges. *Multimed. Tools Appl.* 51 (2), 739–777.
- Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R., 1990. Indexing by latent semantic analysis. *J. Am. Soc. Inform. Sci.* 41 (6), 391–407.
- Demirkus, M., Garg, K., Guler, S., 2010. Automated person categorization for video surveillance using soft biometrics. In: Biometric Technology for Human Identification VII.
- Denman, S., Fookes, C., Bialkowski, A., Sridharan, S., 2009. Soft-biometrics: unconstrained authentication in a surveillance environment. In: Techniques and Applications, Digital Image Computing, pp. 196–203.
- Gallagher, A.C., Chen, T., 2009. Understanding images of groups of people. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 256–263.
- Gao, W., Ai, H., 2009. Face gender classification on consumer images in a multiethnic environment. In: IEEE International Conference on Biometrics, pp. 169–178.

- Golomb, B.A., Lawrence, D.T., Sejnowski, T.J., 1990. Sexnet: a neural network identifies sex from human faces. In: Advances in Neural Information Processing Systems, pp. 572–577.
- Guo, G., Dyer, C., Fu, Y., Huang, T., 2009. Is gender recognition affected by age? In: International Conference on Computer Vision Workshops, pp. 2032–2039.
- Gutta, S., Wechsler, H., Phillips, P.J., 1998. Gender and ethnic classification of face images. In: International Conference on Face and Gesture Recognition, pp. 194–199.
- Helten, F., Fischer, B., 2004. What do people think about CCTV? Findings from a Berlin survey. *Urban Eye* 13, 1–52.
- Hu, S.D., Jou, B., Jaech, A., Savvides, M., 2011. Fusion of region-based representations for gender identification. In: International Joint Conference on Biometrics.
- Jain, A., Huang, J., 2004. Integrating independent components and linear discriminant analysis for gender classification. In: IEEE International Conference on Automatic Face and Gesture Recognition, pp. 159–163.
- Jain, A.K., Park, U., 2009. Facial marks: soft biometric for face recognition. In: IEEE International Conference on Image Processing, November, pp. 37–40.
- Jain, A.K., Dass, S.C., Nandakumar, K., 2004a. Can soft biometric traits assist user recognition? In: Proceedings of SPIE, vol. 5404, pp. 561–572.
- Jain, A.K., Dass, S.C., Nandakumar, K., 2004b. Soft biometric traits for personal recognition systems. In: International Conference on Biometric Authentication, pp. 731–738.
- Jain, A.K., Nandakumar, K., Lu, X., Park, U., 2004c. Integrating faces, fingerprints, and soft biometric traits for user recognition. In: BioAW, vol. LNCS 3087, pp. 259–269.
- Khan, A., Majid, A., Mirza, A.M., 2005. Combination and optimization of classifiers in gender classification using genetic programming. *Int. J. Knowl.-Based Intell. Eng. Syst.* 9, 1–11.
- Kim, H.-C., Kim, D., Ghahramani, Z., Bang, S.Y., 2006. Appearance-based gender classification with gaussian processes. *Pattern Recogn. Lett.* 27, 618–626.
- Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K., 2009. Attribute and simile classifiers for face verification. In: International Conference on Computer Vision, pp. 365–372.
- Lian, X.-C., Lu, B.-L., 2008. Gender classification by combining facial and hair information. In: Advances in Neuro-Information Processing. Springer-Verlag, pp. 647–654.
- Li, S.Z., Chu, R., Liao, S., Zhang, L., 2007. Illumination invariant face recognition using near-infrared images. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 627–639.
- Lu, X., Jain, A.K., 2004. Ethnicity identification from face images. In: Proceedings of SPIE, vol. 5404, pp. 114–123.
- Mahalingam, G., Kambhamettu, C., 2011. Can discriminative cues aid face recognition across age? In: IEEE International Conference on Automatic Face and Gesture Recognition, pp. 206–212.
- Makinen, E., Raisamo, R., 2008. Evaluation of gender classification methods with automatically detected and aligned faces. *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (3), 541–547.
- Marcialis, G.L., Roli, F., Muntoni, D., 2009. Group-specific face verification using soft biometrics. *J. Vis. Lang. Comput.* 20 (2), 101–109.
- Moghaddam, B., Yang, M.-H., 2002. Learning gender with support faces. *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (5), 707–711.
- Niinuma, K., Park, U., Jain, A.K., 2010. Soft biometric traits for continuous user authentication. *IEEE Trans. Inform. Forensics Secur.* 5 (4), 771–780.
- Nixon, M.S., Carter, J.N., 2006. Automatic Recognition by Gait. Proc. IEEE 94 (11), 2013–2024.
- Papadimitriou, Raghavan, P., Tamaki, H., Vempala, S., 2000. Latent Semantic indexing: a probabilistic analysis. *J. Comput. Syst. Sci.* 61 (2), 217–235.
- Park, U., Jain, A.K., 2010. Face Matching and Retrieval Using Soft Biometrics. *IEEE Trans. Inform. Forensics Secur.* 5 (3), 406–415.
- Prince, S.J.D., Aghajanian, J., 2009. Gender classification in uncontrolled settings using additive logistic models. In: International Conference on Image Processing, pp. 2557–2560.
- Reid, D.A., Nixon, M.S., 2010. Imputing human descriptions in semantic biometrics. In: Proceedings of the 2nd ACM Workshop on Multimedia in Forensics, Security and, Intelligence, pp. 25–30.
- Ross, A., Chen, C., 2011. Can gender be predicted from near-infrared face images? In: International Conference on Image Analysis and Recognition, pp. 120–129.
- Ross, A.A., Nandakumar, K., Jain, A.K., 2006. Handbook of Multibiometrics. Springer.
- Samangooei, S., 2010. Semantic biometrics. Ph.D. Thesis, University of Southampton.

- Samangooei, S., Nixon, M.S., 2010. Performing content-based retrieval of humans using gait biometrics. *Multimedia Tools Appl.* 49 (1), 195–212.
- Samangooei, S., Guo, B., Nixon, M.S., 2008. The use of semantic human description as a soft biometric. In: 2nd IEEE International Conference on Biometrics: Theory, Applications and Systems, September, pp. 1–7.
- Shan, C., 2010. Gender classification on real-life faces. In: Advanced Concepts for Intelligent Vision Systems, pp. 323–331.
- Shutler, J., Grant, M., Nixon, M.S., Carter, J.N., 2002. On a large sequence-based human gait database. In: Proceeding of RASC, Springer Verlag, pp. 66–72.
- Socolinsky, D.A., Selinger, A., 2002. A comparative analysis of face recognition performance with visible and thermal infrared imagery. In: International Conference on Pattern Recognition, vol. 4, pp. 217–222.
- Sun, Z., Bebis, G., Yuan, X., Louis, S.J., 2002. Genetic feature subset selection for gender classification: a comparison study. In: Proceedings of Applications of Computer Vision, pp. 165–170.
- Sun, N., Zheng, W., Sun, C., Zou, C., Zhao, L., 2006. Gender classification based on boosting local binary pattern. In: Advances in Neural Networks, pp. 194–201.
- Toews, M., Arbel, T., 2009. Detection, localization, and sex classification of faces from arbitrary viewpoints and under occlusion. *IEEE Trans. Pattern Anal. Mach. Intell.* 31 (9), 1567–1581.
- Wang, J., Li, J., Yau, W., Sung, E., 2010. Boosting dense SIFT descriptors and shape contexts of face images for gender recognition. In: Computer Vision and Pattern Recognition Workshops, pp. 96–102.
- Webster, C.W.W., 2009. CCTV policy in the UK: reconsidering the evidence base. *Surveill. Soc.* 6 (1), 10.
- Xia, B., Sun, H., Lu, B.-L., 2008. Multi-view gender classification based on local gabor binary mapping pattern and support vector machines. In: International Joint Conference on Neural Networks, pp. 3388–3395.
- Yang, Z., Ai, H., 2007. Demographic classification with local binary patterns. In: International Conference on Biometrics, pp. 464–473.
- Zhang, B., Zhang, L., Zhang, D., Shen, L., 2010. Directional binary code with application to PolyU near-infrared face database. *Pattern Recogn. Lett.* 31, 2337–2344.
- Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A., 2003. Face recognition: a literature survey. *ACM Comput. Surv. (CSUR)* 35 (4), 399–458.

A User Behavior Monitoring and Profiling Scheme for Masquerade Detection

Ashish Garg¹, Shambhu Upadhyaya¹, and Kevin Kwiat²

¹*Department of Computer Science and Engineering,*

State University of New York at Buffalo, Buffalo, NY 14260, USA

²*Air Force Research Laboratory, 525 Brooks Rd., Rome, NY 13441, USA*

Abstract

Masquerading attack refers to conducting malicious activities on a computer system by impersonating another user. Such attacks are difficult to detect with standard intrusion detection sensors when they are carried out by insiders who have the knowledge of the system. One approach to detect masquerading attacks is to build user profiles and monitor for significant changes in user's behavior at runtime. Intrusion detectors based on this principle typically have used user command line data to build such profiles. This data does not represent user's complete behavior in a graphical user interface (GUI)-based system and hence is not sufficient to quickly and accurately detect masquerade attacks. In this chapter, we present a new empirically driven framework for creating a unique feature set for user behavior monitoring on GUI-based systems. For proof-of-concept demonstration, we use a small set of real user behavior data from live systems and extract parameters to construct these feature vectors. The feature vectors contain user information such as mouse speed, distance, angles, and amount of clicks, and keystroke dynamics during a user session. We then formulate our technique of user identification and masquerade detection as a *binary classification problem* and use Support Vector Machine (SVM) to learn and classify user actions as intrusive or benign. We show that our technique based on these feature vectors can provide detection rates of up to 96% with low false positive rates. We have tested our technique with various feature vector parameters and concluded that these feature vectors can provide unique and comprehensive user behavior information and are powerful enough to detect masqueraders.

Keywords: GUI-based profiling, intrusion detection, masquerade detection, support vector machine

1. Introduction

Masquerading is a process where a person spoofs someone else's identity and utilizes privileges to which he is not entitled to. Masquerading could be quite damaging in that the masquerader could steal important personal, and organizational information as well as the identity of the victim. In the computer security domain, a masquerade attack can be described as follows. During the absence of a legitimate user, the attacker gets access to his computer, either due to the fact that the victim left his terminal open or because the attacker somehow learned of his password or the attacker is able to exploit a system vulnerability and escalate his privilege. Masquerading can be performed by insiders or outsiders. Masquerade detection becomes difficult under the circumstances when the masquerader is very familiar with the victims' behavior and can mimic it to a large extent. An *insider masquerader* is a legitimate user for the system and can perform a lot more actions without being detected for long periods of time as compared to his *outsider* counterpart. According to the CSI Computer Crime and Security Surveys (Richardson, 2008), the collected data shows that along with the virus and worm attacks, the insider threats are also rising. An outsider on the other hand can gain root access to a victim's machine remotely and steal information and cause damage to the system.

The most useful information which can be used to detect masquerade attacks is contained in the actions a masquerader performs. This set of actions is known as the *behavior profile*. Masquerade detection techniques are based on the premise that when a masquerader attacks the system, he will sufficiently deviate from the victim's behavior profile and thus be caught. To be able to make a distinction between normal and malicious behavior, these detection systems collect data from user sessions and build baseline profiles. This data is initially used to train the detection systems about what is *normal* and later for testing the detection of the *malicious* activity.

Masquerade detection which is based on the principle of anomaly detection (Porras and Neumann, 1997) can be performed at various levels such as network packets, system calls (program execution time), and user command level. In almost all Unix-like systems, most of the user activities are performed by typing commands on a terminal. These activities include opening of new applications, closing them, and managing other computer processes. Most of the existing Intrusion Detection Systems (IDS) base their detection decision on this *command line-based* dataset. There have been several attempts to solve the masquerade detection problem by utilizing Unix command line sequences derived from this dataset (Schonlau et al., 2001a; Maxion and Townsend, 2002). Although this dataset can provide effective anomaly detection capabilities in these systems, they are not sufficient to provide anomaly detection in graphical user interface (GUI)-based environments like Windows operating system and some variants of Unix (e.g., Linux, Mac OS, etc.). The reason for this inadequacy is that user's behavior in GUI-based systems is vastly different from that of command line-based systems. In a GUI-based system, most of the user activities are performed using either mouse movement and clicks or a combination of mouse movement and keystrokes. The command line data cannot capture this GUI-based behavior and further efforts are required to incorporate this behavior information for effectively detecting anomalies.

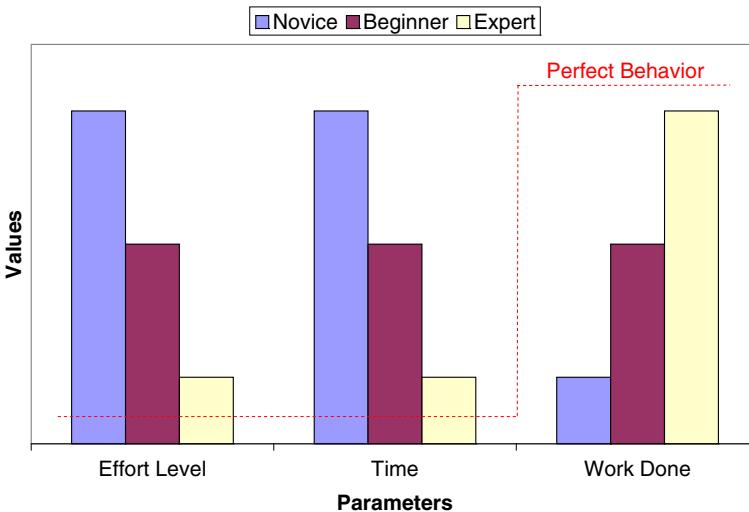


Fig. 1. Behavioral path for various types of users.

To further motivate the problem, let us look at the values of effort level, time taken to complete the task and the work done for various types of users such as novice, beginner, and expert. Figure 1 shows the general trend of these types of users. The values of various parameters are drawn based on users' activities on the system. These activities include but are not limited to, mouse movements, keyboard typing speed, number of commands executed, and the duration taken to perform a job function.

As can be seen from the pictorial representation, the behavior of users differs greatly depending on their skill levels. An expert user will be more toward the *perfect behavior* statistics because of his knowledge and experience of using the system. The perfect behavior is defined as the minimum effort level or system activity involved in getting the maximum work done in the shortest possible time. A novice user shows a greater variation in the behavior due to lack of experience and knowledge about the applications being used on the system. The parameter values in the figure for effort level, time taken to complete the task, and work done during the time are indicative of this factor. For example, an expert's behavior will show shortest time to complete the job along with fewest steps (in terms of commands, mouse clicks, etc.). It should be noted that although this deviation can come close to the perfect behavior line, it will not touch it ideally. It is also interesting to note here that for performing the same job function various users will utilize the system differently, thus generating a totally different behavior profile. This leads to the fact that carefully selected behavior features of various users can provide details about their behavior profiles and thus enable a more accurate anomaly detection as compared to utilizing the command line-based audit data alone.

In this chapter, a new framework is presented for collecting GUI-based user behavior characteristics such as mouse movements and clicks, typing speed, and system background processes and utilizing it to create user behavior profiles.

Relevant mouse- and keyboard-related features such as average distance, speed, angles of movement, and number of clicks, typed text, and commands during a session are extracted. Support Vector Machines (SVM) are utilized to learn the user specific feature sets and to show that these sets provide unique characteristics for identifying a user and thus will help in identifying the masqueraders faster and much easier. First, the unique features are selected and are utilized to construct feature vectors. The feature vectors are then used to create $\langle \text{feature}: \text{value} \rangle$ tuples for the support vector machine software. The training and classification of the support vector machines is done using these feature vectors and the classification results are reported and analyzed.

1.1. Summary of contributions and chapter organization

This is the first work that incorporates a comprehensive set of GUI features for masquerade detection and performs a detailed analysis. The specific contributions are:

- An empirically driven framework for creating a unique and comprehensive set of feature vectors in a GUI-based system.
- A profile creation tool to generate user behavior profiles from the extracted features.
- A proof-of-concept demonstration by utilizing the Support Vector Machine (SVM) technique to train the systems and test detection accuracy.

The rest of the chapter is organized as follows. Section 2 reviews related work and Section 3 gives some background on SVMs. Section 4 gives the details of data collection, feature extraction, and feature vector generation for our experiments. The experimental design is given in Section 5 and a discussion of results and the conclusion appear in Section 6.

2. Related work

Schonlau (1998) collected Unix command line data of 50 users for testing and comparing various intrusion detection methods. This dataset is useful for representing user profiles for users in a Unix-like system where most of the user activity is recorded as the command sequence. However, it would not be able to accurately represent the behavioral profiles of users working on modern graphical user interface (GUI)-based operating systems such as Microsoft Windows and some variants of Unix (such as Linux and Mac OS). This is due to the fact that GUI-based systems are designed to be more visual and users are expected to use mouse clicks for activating commands rather than typing a command on the terminal. Command line data is not capable of capturing the user's actions such as mouse movements, keyboard typing speed, and any mannerisms, which could enhance the accuracy of correctly determining user's behavior on these systems.

Our work finds great relevance in areas that utilize user behavior data to make certain decisions about users. Specifically the domain of application for this work would be the masquerade detection techniques. Therefore we review some

well-known works in this area and identify the recent improvements in the intrusion detection techniques. We will then discuss it in the context of user behavior modeling and profile generation.

2.1. Previous anomaly/masquerade detection efforts

User behavior-based anomaly detection systems have been proposed and developed by many research groups. Most of the user profiles are generated based on command line data. One of the first to introduce the behavioral profiles for intrusion detection was Denning and Neumann (1985). Lane et al. (1997a,b) assert that user actions are causal in nature and sequence matching can provide appropriate solutions for detecting anomalies. They show that a Hidden Markov Model can provide a good approximation of user command behavior (Lane, 1999). In another work, Schonlau et al. (2001b) utilized various statistical techniques (“Uniqueness,” “Bayes one-step Markov,” “Hybrid multi-step Markov,” “Compression,” “IPAM,” and “Sequence Match”) for evaluating their effectiveness in masquerade detection. Naïve Bayes Classifier was used by Maxion and Townsend (2002) on a truncated user command dataset. They provide results to prove that their technique improves detection significantly giving very low false positive rates. In Maxion (2003) they claim that enriched command dataset results in a better detection accuracy. In a later work (Maxion and Townsend, 2004) they provide a new classification algorithm to improve the results obtained earlier using Schonlau’s data. Wang and Stolfo (2003) show that one-class SVM training works as good as multi-class training for masquerade detection, requiring much less data and more efficient training.

As far as the use of GUI-based data for intrusion detection as well as user authentication techniques is concerned, there has been some effort in the area of keystrokes dynamics (Gunetti and Picardi, 2005; Bergadano et al., 2002; Monroe and Rubin, 1997; Shavlik et al., 2001; Gupta et al., 2008), mouse movements (Goecks et al., 1999; Pusara and Brodley, 2004), and a combination of both keystroke and mouse dynamics (Ahmed and Traore, 2005). Revett et al. (2008) survey user authentication approaches using mouse movements. Other user behavior profiling methods for detection used system call monitoring (Hofmeyr et al., 1998; Lee et al., 1999; Forrest et al., 1997; Warrender et al., 1999); analysis of audit logs (Ilgun et al., 1995; Li et al., 2002; Javitz and Valdes, 1991; Wespi et al., 2000; Ye, 2000); and program execution traces (Ghosh et al., 1999; Levitt and Ko, 1994; Marceau, 2000; Michael and Ghosh, 2000; Wagner and Dean, 2001; Rajagopalan et al., 2002). Lastly, Feng et al. (2003) proposed anomaly detection using call stack information. These approaches either rely on smaller set of GUI-based information or use this GUI data for purposes other than masquerade detection, such as user authentication. To be able to make an informed decision about masquerading attempts on a system, the use of comprehensive data from GUI-based systems is essential.

In this chapter, some ideas from the above-mentioned efforts are leveraged and it is shown that the GUI-based data can be used to extract behavior profile features and uniquely identify users, providing better capabilities for detecting masquerade attacks.

2.2. *Masquerade detection based on GUI data*

User behavior represents relatively higher level activities a user performs by physically interacting with a computer. For instance, these activities could be in the form of commands a user runs on the system, his habits in using a particular feature of the system (keystroke combination or typing command completely) and the degree of nervousness while using the computer. As noted in the previous section, the skill level of a particular user will determine the quality of these interactions. A novice will use the computer differently as compared to an expert user and thus generate a different set of behavioral characteristics.

There have been various attempts to use the GUI-based information for profile building and anomaly detection. These techniques were either used for re-authenticating the user using his mouse movement characteristics (Pusara and Brodley, 2004) for masquerade detection, or for automatically ranking the web pages on a site based on the mouse clicks and navigation features (Brown and Claypool, 2003; Claypool et al., 2001), or for improving the effectiveness of interface design for providing the dynamic content on web pages (Mueller and Lockerd, 2001). These techniques were either very specific to an application or were presented in the context of a limited set of characteristics, making them insufficient for use with masquerade detection on a host for all kinds of activities. We presented a preliminary framework (Garg et al., 2006) for developing user behavior profiles for GUI-based systems for use with masquerade detection systems. In this chapter, we provide a comprehensive set of features to be used for profile building to improve the detection accuracy.

A combination of keystroke dynamics and 2D face recognition has been explored in Giot et al. (2010) but this technique requires the deployment of additional hardware. Imsand et al. (2007, 2008) created user profiles based on how the windows, icons, menus, and pointers, that comprise a graphical user interface, are manipulated by individual users. The authors used Jaccard Index as well as Artificial Neural Networks for identification of users. This approach requires physical characteristics (Age, Gender, Ethnicity, Hair Color, etc.) of the people, which may be difficult to obtain due to the nature of the data, thus rendering it inapplicable in a generic setting.

3. Support Vector Machines (SVMs)

Support Vector Machines (SVMs) are a set of related supervised learning methods used for classification and regression (Wikipedia, 2012). SVMs were developed by Vapnik (1995,1998), based on the structural risk minimization principle (Vapnik, 1982) from statistical learning theory. As described in Joachims (2002), the idea of structural risk minimization is to find a hypothesis h from a hypothesis space H for which one can guarantee the lowest probability of error $Err(h)$ for a given training sample S_n represented as:

$$(\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n) \quad \vec{x}_i \in \Re^N, \quad y_i \in \{-1, +1\} \quad (1)$$

of n examples. Based on this input, SVMs can learn the linear decision rules $h(\vec{x}) = \text{sign}\{\vec{w} \cdot \vec{x} + b\}$ described by a weight vector \vec{w} and a threshold b' .

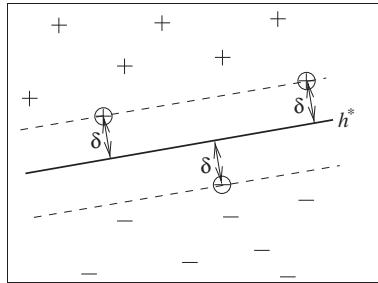


Fig. 2. A two-dimensional binary classification problem for support vector machines (Joachims, 2001).

3.1. Linear SVMs

If the training set is linearly separable, at least one hyperplane can be found, such that all the positive training examples are on one side of the hyperplane, while all negative training examples lie on the other. This requirement can be stated as

$$y_i[\vec{w} \cdot \vec{x}_i + b] > 0 \quad (2)$$

for each training example (\vec{x}_i, y_i) . The SVM finds the hyperplane by calculating the maximum Euclidean distance to the nearest training example. This distance is represented by δ in Fig. 2. The nearest examples to the hyperplane, shown in the figure as circles, are called support vectors (SVs).

The problem of finding the maximum margin hyperplane can be translated into the following optimization problem:

Optimization Problem 1 (HARD-MARGIN SVM (PRIMAL))

$$\text{minimize : } V(\vec{w}, b) = \frac{1}{2} \vec{w} \cdot \vec{w}, \quad (3)$$

$$\text{subject to : } \forall_{i=1}^n : y_i[\vec{w} \cdot \vec{x}_i + b] \geq 1. \quad (4)$$

Constraint (4) states that all the training examples should lie on the correct side of the hyperplane. It should be noted that

$$\delta = \frac{1}{\|\vec{w}\|}. \quad (5)$$

In case the training examples are not linearly separable, the training fails. This case is solved by using *soft-margin* SVM. Soft-margin SVMs include an upper bound on the number of training errors in the objective function of Optimization Problem 1. This upper bound and the length of the weight vector are then both minimized simultaneously.

Optimization Problem 2 (SOFT-MARGIN SVM (PRIMAL))

$$\text{minimize : } V(\vec{w}, b, \vec{\xi}) = \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum_{i=1}^n \xi_i, \quad (6)$$

$$\text{subject to : } \forall_{i=1}^n : y_i[\vec{w} \cdot \vec{x}_i + b] \geq 1 - \xi_i, \quad (7)$$

$$\forall_{i=1}^n : \xi_i \geq 0. \quad (8)$$

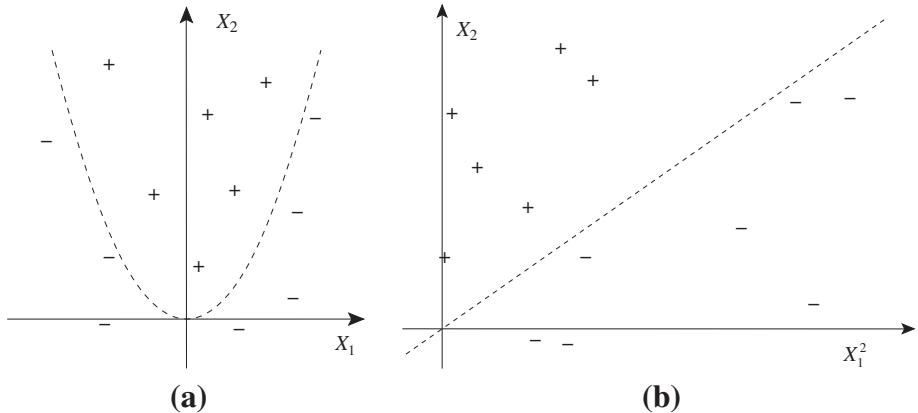


Fig. 3. Transformation of SVMs to nonlinear learners (Joachims, 2002).

The ξ_i are called *slack* variables. When the training example lies on the *wrong* side of the hyperplane, the corresponding ξ_i is greater than 1. $\sum_{i=1}^n \xi_i$ is the upper bound on the number of training errors and factor C in Eq. (6) allows trade-off between training error vs. model complexity. Small values of C increase the training errors, while larger values bring it closer to the hard-margin SVM.

3.2. Nonlinear SVMs

Complex real-world problems have nonlinear structure, thus making the linear classifiers inappropriate for use. SVMs can be easily transformed into nonlinear learners (Boser et al., 1992). This is done by mapping the attribute vectors \vec{x}_i into a high-dimensional feature space X' using a nonlinear mapping $\Phi(\vec{x}_i)$. Maximum-margin linear classification rule is learned in the feature space X' . Figure 3a shows a training set that is not linearly separable in (x_1, x_2) . Figure 3b shows the same problem after the nonlinear transformation. It should be noted that although such a mapping $\Phi(\vec{x})$ is inefficient to compute, using the special property of SVMs as defined in Boser et al. (1992), it is sufficient to compute dot-products in the feature space, i.e., $\Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j)$ during the training and testing. Such dot-products can be computed efficiently using *kernel* functions $\kappa(\vec{x}_1, \vec{x}_2)$. It can be written as

$$\Phi(\vec{x}_1) \cdot \Phi(\vec{x}_2) = \kappa(\vec{x}_1, \vec{x}_2). \quad (9)$$

Various kernels can be defined for SVMs as polynomial classifiers, radial basis function (rbf), or two-layer sigmoid neural nets

$$K_{\text{poly}}(\vec{x}_1, \vec{x}_2) = (\vec{x}_1 \cdot \vec{x}_2 + 1)^d, \quad (10)$$

$$K_{\text{rbf}}(\vec{x}_1, \vec{x}_2) = \exp(-\Gamma(\vec{x}_1 - \vec{x}_2)^2), \quad (11)$$

$$K_{\text{sigmoid}}(\vec{x}_1, \vec{x}_2) = \tanh(s(\vec{x}_1 \cdot \vec{x}_2) + c). \quad (12)$$

3.3. Advantages of SVMs

Support Vector Machines (SVM) are efficiently used for learning and classifying users as reported in Wang and Stolfo (2003). They are used in our research to learn a rich feature set for GUI-based user behavior and classify various users for masquerade detection. The main advantages of using SVM in the context of user behavior learning and masquerade detection are:

- SVMs are maximal-margin classifiers as compared to, say, Naïve Bayes, which is probabilistic.
- They have been known to be highly effective in text classification (Joachims, 1998) and scale well for the rich features, which is essential in GUI-based datasets.
- SVMs can provide better classification results with less training data.
- One user's learning does not depend on other users data. Stated another way, one user's profile can be built using his own data without utilizing other user's data (Wang and Stolfo, 2003).

The feature set for GUI-based behavior characteristics is tested with linear kernel as well as radial basis function kernel, thus following the Eqs. (3), (6), and (11). When using SVMs for masquerade detection, a binary classification problem is defined due to the fact that users will be tagged as positive (genuine) or negative (masquerader).

4. Data collection, feature extraction, and feature vector generation

4.1. Data collection

Real user behavior data for multiple users has been collected and unique parameters are extracted to construct the feature vectors. For this purpose, an *active system logger* has been developed using the Microsoft .NET framework and C# language on Windows XP system. The .NET framework has been chosen due to its ease of use and ability to interact with various Windows components seamlessly. This logger is designed such that it is able to collect system events due to all possible user activities on the system in real time. The logger collects events such as keyboard activity, mouse movement coordinates and mouse clicks, system background processes, and user run commands. Table 1 shows a sample of the data collected by the logger. As can be seen from the table, the logger collects the timestamp information along with the events, such as mouse coordinates, mouse clicks, process information, and keyboard statistics.

GUI-based parameters such as mouse movements and keystroke activities require the knowledge of the context in which they are being executed. This context information is useful in fine-tuning the parameters required for behavior modeling and helps in eliminating false positives during detection if used at the training and testing stages of an IDS. The background processes currently being run along with the processes that have been started while the data is logged are an important resource in precisely determining the changes in the system due to user activities and can be used for improving detection performance.

Table 1
A sample dataset collected by the logger

CPU ticks	Event	Value
632784114615719648	Mouse coordinates	666,621
632784114615719648	Left click	
632784116067707504	Keypress	<i>n</i>
632784118238474928	Right click	
632784120702271680	New process	notepad.exe
632784120728909984	Process terminate	notepad.exe

4.1.1. Issues with data collection and use

The data collection process is a challenging task and involves many issues that must be addressed before the data is collected and used. The main issues in the process of data collection and utilization are:

- It is a tedious job and takes a lot of time ranging from weeks to months as reported in [Lane and Brodley \(1999\)](#).
- It is obtrusive and involves user privacy issues, among other problems.
- The collected data may not be in the usable form right away and may require additional efforts to make it usable.

It should be noted that although it is possible to purchase off-the-shelf software to log various GUI-based activities, such a tool will not be easily customizable for specific features for training and testing purposes. Also, data from real users from such tools is not easy to obtain and is not readily available in the literature for research purposes. The tool developed by us for this research is the first such effort in this area. The benefits of obtaining such an extensive dataset are the following:

- It can provide information about *how* the actions are being performed as opposed to *what* actions are performed, and *which* command line provides this.
- It is closest to the user, because it provides the most comprehensive information about user activity on a system.

4.1.2. Data sanitization

It should be noted that the data collected using the logger can be huge due to the fact that all the system activities are being captured. Also, the mouse movement data is captured for each CPU tick; it comprises of a large portion of log files. Data sanitization is done by removing the following information:

- User credentials: user's private information is removed to protect their identity.
- Mouse coordinates not leading to an event. An event is described as either start or termination of an application or a useful click in terms of minimizing or maximizing a window.
- Timestamps associated with the mouse clicks. Only the number of clicks in a certain period is considered.

4.2. Feature selection methods

There have been many feature selection methods around in many areas of research such as machine learning, data mining, and clustering techniques and in some cases where statistical deviation is to be determined. The most common method of feature selection utilizes *weighting measures*. The weighting measure feature selection can be done by two methods. In the first method, highest weighted features are calculated and based on the number of features to be retained or discarded, a threshold is determined. Any features below the threshold are discarded. The advantage of this method is that depending on the type of the classification engine being used, a suitable number of features can be selected. This method also helps in determining the threshold, which is data dependent. The second method involves determining a reasonable threshold based on the type of data and the application of results, and then discarding any features which are below the threshold. The advantage of the second method is that only the features which fulfill the requirements of threshold are kept and used for classification, thus giving a better precision and more confidence in results.

Numerous measures have been presented in the past for weighting (or scoring) the features. These include *information gain*, *odds ratio*, χ^2 , and *term strength*. It is important to note that the feature weighting and selection process are coordinated with the classification model being used (Brank et al., 2002). For a GUI-based user behavior data, the feature selection poses several challenges. The selection process in such a case depends on the amount of collected data, session duration, type of user activities involved, and the screen resolution. On the one hand, a user with more text typing task will produce a keystroke-rich dataset and hence the feature selection as well as the threshold for keystroke features will be higher than that for mouse-related activities. On the other hand, for a graphics designer, the use of mouse activities will be far more than the keystroke activities and hence the threshold set for mouse events will be higher and more mouse features should be selected to improve the confidence with which the classification results are calculated.

In this chapter, feature selection has been done mainly based on the fact that GUI-based user behavior is significantly different from that in a traditional command-line-based system. However, it is necessary to look carefully into the GUI specific data in the collected logs to select appropriate features for building the behavior model. The next section presents the issues in feature extraction and describes the process used for parsing the collected and sanitized data to extract useful features.

4.3. Feature extraction

After the sanitization is done on the collected GUI-based user behavior data, the feature extraction is done based on the various methodologies relevant to the specific type of *data segment*. A data segment is the portion of collected data which belongs to a particular user behavior characteristic. For example, the collected data related to mouse activities on a system is considered a data segment.

The feature extraction process is shown in Fig. 4. Initially, the logger collects the real user behavior data from a Windows XP system. These logged data files consist of various GUI behavior features such as executed commands, mouse clicks, mouse movement details, keyboard activities, context information, and the background

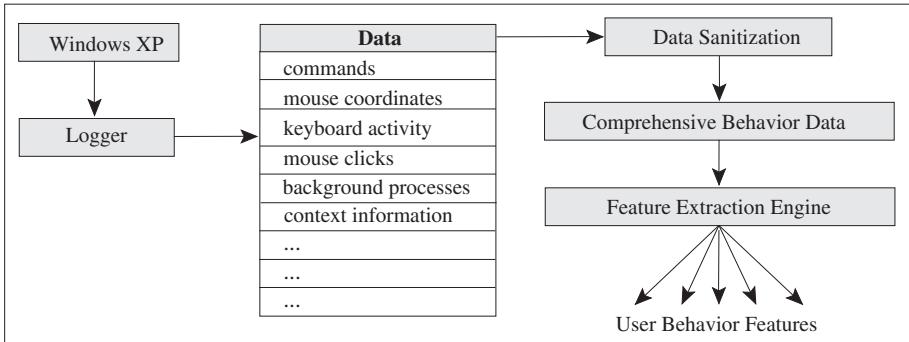


Fig. 4. Feature extraction process for logged data.

process information. This data is then sanitized to remove any user identifiable information such as username and typed passwords. After the sanitization is done, the comprehensive data is used to extract useful features. A feature extraction engine has been developed to extract these GUI-based user behavior features. The following measurable features are extracted using the engine:

- *Mouse clicks (lc and rc)*: The average number of *left* and *right* mouse clicks per user session as well as activity for each 10 min window during the session. The 10 min activities are recorded to improve the confidence in the value of this feature and to eliminate any errors due to any inconsistency.
- *Distance (d)*: The average distance traveled by mouse per event. Examples of such events include closure of a window, start of a process, and process termination. This is calculated by using a small distance as a scale to measure the total distance covered by the mouse for an event. This method of distance measurement is more advantageous in providing the real distance covered. The point-to-point measurement, which relies only on the start and end position of mouse, provides only the linear distance.
- *Mouse speed (s)*: The average speed of movement for the entire session as well as for events. This is calculated by dividing the distance calculated above by the time taken for the event.
- *Mouse movement angle per event (θ values)*: The angles of mouse movement relative to the x-axis for each event. The angles θ_{si} and θ_{ti} for starting position of mouse for an event and the end position or a click are calculated, respectively. These θ values for the mouse events are calculated as shown in Fig. 5. Due to the fact that these events can be reached at from either left or right directions, there is a total of four angles (two each for *left* and *right*), namely θ_{sl} , θ_{rl} , θ_{sr} , and θ_{tr} . The shaded dots represented at the four corners in the figure indicate the *start* or *termination* of an event.
- *Keystroke time*: The time duration for which a particular key has been kept pressed is called *keystroke time*. It is calculated by measuring the time between the *KeyDown* and *KeyUp* events collected by the logger for $[a-z]$, $[A-Z]$, $[0-9]$, and special characters (*Ctrl*, *Alt*, *?*, *@*, etc.). It should be noted that some of the special

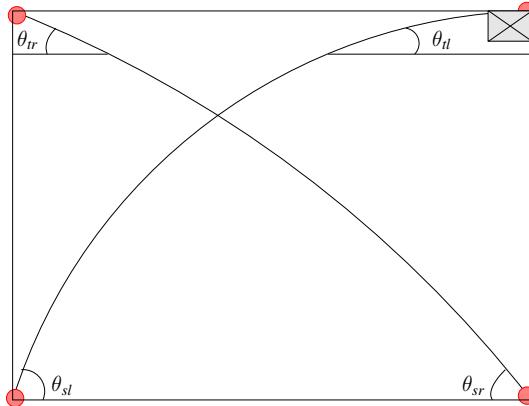


Fig. 5. Calculations of angles associated with mouse events.

character keys are associated with a regular character such as *LShiftKey + A*, in which case the combined time is considered.

- *n-graph latency*: *n*-graph latency represents the time delay between the *KeyDown* event of first key and the *KeyUp* event of the *n*th key. It is calculated for *di-graph*, *tri-graph*, *quad-graph*, and *penta-graphs*. These values are calculated by using *R-measure* for the dataset based on the method provided in [Gunetti and Picardi \(2005\)](#). The idea of R-measure was introduced by [Bergadano et al. \(2002\)](#).

Apart from these *raw* features, the *mean* “*m*” and *standard deviation* “*sd*” for all the raw features are also calculated.

4.3.1. Sliding window protocol

During a logged session, there may be certain period of inactivity or a break taken by the user. These inactive periods can cause errors in the calculation of mouse clicks for the session and appropriate steps must be taken to avoid these errors. A window of 10 min has been chosen and the mouse clicks during all the 10 min windows have been calculated. To balance out the errors caused during the borderline cases of window activity (that is during an active session), a sliding window protocol has been devised as follows:

Initially, the mouse clicks (*lc* and *rc*) per 10 min of activity for the entire session are calculated. If the session starting time *t* = 31, then the number of clicks for a 10 min period following 31 are calculated (first period would be [31 – 41] and second will be [41 – 51] and so on). After finishing this iteration, the window is then slid to time *t* = 32 and the number of clicks for the following period is calculated such that the first period would be the range [32 – 42], second will be [42 – 52], and so on. The values for 10 iterations are then calculated. After 10 iterations the pattern repeats itself and no more calculation is required. This gives 10× values, i.e., values from 10 slides of windows for the entire session. Figure 6 shows a generalized case for this sliding windows protocol. The advantage of this protocol to calculate values would

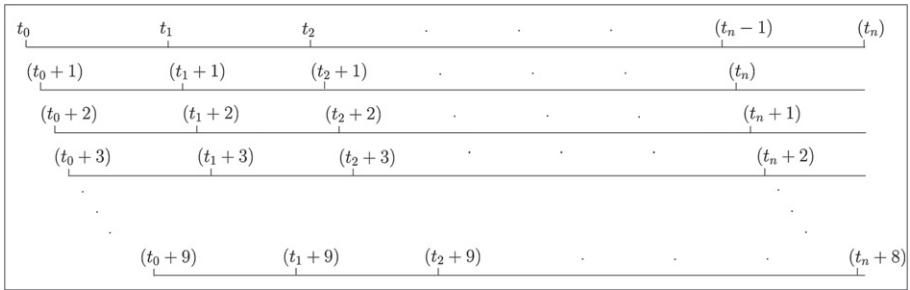


Fig. 6. Sliding window protocol.

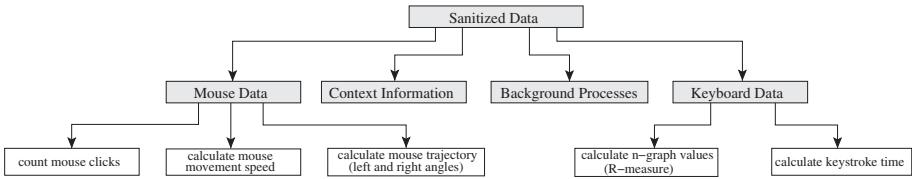


Fig. 7. Feature extraction engine flow diagram.

be clear when various techniques for learning and classifying behavior features are described in Section 5.

4.3.2. Design of feature extraction engine

The feature extraction engine extracts the useful GUI-based user behavior features from the logged data. This tool has been developed in Windows using Java programming language and provides components for extracting mouse as well as keystroke-related features. The flow diagram in Fig. 7 illustrates the various components of this engine.

The mouse activities for the user are calculated as follows. The total number of mouse clicks (left as well as right clicks) during a session is counted, while also keeping a count of these clicks during each 10 min period. For each successful event in the log data, the total distance covered by the mouse is calculated. This distance is used to calculate the average mouse speed during the event. The mouse angles for these events are also calculated for the start of event as well as the end.

The keystroke data is parsed to calculate the keystroke time for all the keystrokes individually using the R-measure method described by [Gunetti and Picardi \(2005\)](#). It should be noted that due to the type of data being logged, the R-measure results for n -gram values above tri-graph tend to provide insufficient results and thus may be discarded. In the case of text intensive datasets, these values may be utilized. The extraction engine is flexible to accommodate this by providing a threshold value, which can be used to filter unnecessary features with low value.

Table 2
Sample SVM tuples

#	Category	(Feature number):(Value)					
+1	6:0.0198403253586671	15:0.0339873732306071	29:0.0360280968798065	31:0.0378103484117687	...		
+1	6:0.00499945516847248	12:0.0674048171189473	26:0.0432780404880932	27:0.0145572144203208	...		
+1	2:0.0218536049648761	6:0.0267664363140353	11:0.0962235717863764	15:0.0366816912369227	...		
-1	6:0.0156041309085692	27:0.136306460498695	41:0.14370264867204	77:0.352992756232454	...		
-1	27:0.297595379637533	41:0.235307505583443	51:0.347070910968236	57:0.38151801281582	...		

4.4. Feature vector generation

In order to effectively utilize the extracted feature set for training and classification purposes, these features are put together in the form of feature vectors. The feature vectors contain the feature number as well as the value associated with that feature. The values of the feature are not deterministic and vary based on the behavior of that particular user. To make the task of feature vector creation easier to understand and improve the precision in training and classification, the feature vector values are normalized before the tuple creation for the SVM training module. The normalization of these features is done using the following equation:

$$N_i = \frac{f_i - f_{\min}}{f_{\max} - f_{\min}}, \quad (13)$$

where N_i is the normalized value of the feature f_i , whereas f_{\min} and f_{\max} are the minimum and maximum values for the calculated feature set.

After normalizing the feature values, a *tuple creation module* is used to create SVM compatible tuples. Table 2 shows a sample set of tuples being created. The +1 represents the positive example whereas -1 represents a negative example for training the SVM. Features f and their corresponding normalized values v are represented in the $(f:v)$ format.

5. Experimental design

This section describes the proof-of-concept experimental design associated with the training and classification of the SVMs with user behavior profiles. In order to run the feature vector tuples, *SVM^{light}* software (Joachims, 2004) has been used, which implements Vapnik's Support Vector Machine (Vapnik, 1995). GUI-based data has been collected from a set of three consenting users in our lab using the logger described in Section 4.1. This data was fed to the parsing engine, as described in Section 4.3 to parse and extract the useful features. The above-mentioned methodology was then utilized to create the SVM tuples. The overall methodology for training and testing using SVMs tuples is as follows:

- Datasets are obtained for three distinct users A, B, and C.
- The obtained dataset is split for training and testing by using $(\frac{2}{3})rd$ and $(\frac{1}{3})rd$ rule, respectively.

Table 3
Training and testing sessions for users A, B, and C

User	Total sessions	Training sessions	Testing sessions
A	27	19	8
B	9	6	3
C	50	35	15

- Using the sliding window protocol as described in Section 4.3.1, $10 \times$ tuples are created.
- The training and testing sets thus created are used by the *SVM^{light}* software as described above for calculating the correct classification rates and false alarms.

In the remainder of this section, two different experiments are described. The first experiment shows the classification results based only on 16 features extracted from mouse-related data, whereas the second experiment takes into account a larger number of features including the mouse as well as the keyboard data to calculate the classification rates. The estimation error (a statistical confidence measure) is also calculated for all the three users for various values of generated support vectors. According to Duda et al. (2001), the estimation error is defined as the error arising from the fact that the parameters are estimated from a finite sample. This error can best be reduced by increasing the amount of training data.

5.1. Experiment I: mouse data

In this experiment, the mouse-related data from three users was considered while extracting the features. As mentioned earlier, the mouse data represents very specific user features related to behavioral psychology, degree of nervousness, moods, etc., and can alone be very useful in identifying the specific user. In the first experiment, we extracted only the mouse-related data to create the user profiles. In the second experiment, we utilized the comprehensive dataset features to measure the user profiles and compare the results. Table 3 shows the count of total number of sessions for these users as well as the number of training and testing sessions chosen for this experiment.

Once the sessions are split, using the sliding windows protocol of Section 4.3.1, $10 \times$ tuples for each session are calculated. These tuples are first fed to the training module and then the classification module of *SVM^{light}* software. The classification results are then calculated for three users. The training/testing for one user is done against the remaining two users ($[A \rightarrow (B, C)]$, $[B \rightarrow (A, C)]$, and $[C \rightarrow (A, B)]$). Here, $[A \rightarrow (B, C)]$ means that A is masquerading B and C. To determine the false alarm rate, a user's actions are tested against self. The results from this test are shown in Table 4.

The classification results are also obtained by varying the number of features from 16 down to 14, 12, and 8 and determining the change in the detection rates. This procedure helps in empirically determining the optimal number of features for the best detection rates. The features corresponding to each set are tabulated in Table 5.

For our evaluation, we use two simple performance metrics, viz. correct classification rate and incorrect classification rate. The correct classification rate

Table 4

Classification rates for users A, B, and C with 600 training and 260 testing samples for 16 features; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	92.31%/240 (KC)	7.69%/20 (IC)	308	≤50.67
B → (A, C)	85.77%/223 (KC)	14.23%/37 (IC)	86	≤13.67
C → (A, B)	96.15%/250 (KC)	3.85%/10 (IC)	166	≤26.67

Table 5

Calculation of features for different sizes

Features	Description
16	$\{(lc, rc, d, s, \theta_{sl}, \theta_{tl}, \theta_{sr}, \theta_{tr}) * (m, sd)\}$
14	$\{(d, s, \theta_{sl}, \theta_{tl}, \theta_{sr}, \theta_{tr}) * (m, sd) + (lc, rc)\}$
12	$\{(\theta_{sl}, \theta_{tl}, \theta_{sr}, \theta_{tr}) * (m, sd) + (lc, rc, d, s)\}$
8	$\{(lc, rc, d, s, \theta_{sl}, \theta_{tl}, \theta_{sr}, \theta_{tr}) * (m)\}$

Table 6

Classification rates obtained by varying the number of features for user C

Features	Correct classification rate (%)	False alerts
16	96.15	10
14	74.23	67
12	73.85	68
8	73.85	68

which accounts for the identification of benign as benign and malicious as malicious is an indicator of detection rate. The incorrect classification rate which accounts for the misclassification of benign as malicious and malicious as benign is an indicator of false positives. As can be seen from Table 6, the correct classification rate goes down as the number of features is reduced. Although some earlier approaches used more features to profile the user (Pusara and Brodley, 2004), it can be observed from the results that 16 is a reasonable number (of features) to make a distinction between multiple users. Also, incorporating more features in the set such as keystroke dynamics and process information may improve the results, which will be discussed in the next section.

The classification rate variation for user A is also tested by changing the values for *gamma* parameter (*rbf kernel*) as described in Eq. (11). It should be noted that the radial basis functions are special class of functions. Their response decreases or increases monotonically with distance from a central point (Orr, 1996). There was an improvement in the correct classification rate as the gamma value was increased from 0.25 to 2 (see Fig. 8).

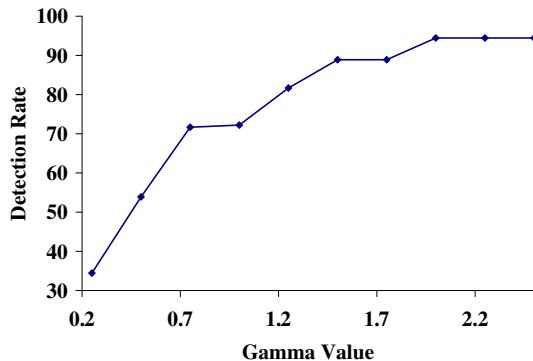


Fig. 8. Classification rate improvement with change in gamma value (rbf kernel) for user A.

Table 7
Training and testing sessions for users A, B, and C

User	Total sessions	Training sessions	Testing sessions
A	27	18	9
B	41	28	13
C	42	28	14

5.2. Experiment II: comprehensive data

After incorporating the keystroke analysis results from the extraction engine, the feature set has been increased from 16 features to a few 100 features and thus a richer set is utilized to perform a second level of classification testing. Various combinations of this rich feature set are utilized to calculate the classification rates as described in the remainder of this section. The feature sets are considered linearly separable and thus the appropriate methods from Section 3 are used to calculate the classification results.

Table 7 shows the count of total number of sessions for these users as well as the number of training and testing sessions chosen for this experiment. It should be noted that the total sessions and the training and testing sessions for this experiment have been changed from the previous experiment due to several reasons. More data has been collected for user B during the experimentation time, increasing the number of total sessions to 41 as opposed to 9 previously. Also, due to the use of comprehensive mouse and keyboard data, some of the sessions of user C were found to be inadequate for profiling and have been discarded.

Once the sessions are split using the sliding windows protocol of Section 4.3.1, $10 \times$ tuples for each session are calculated. These tuples are first fed to the training module and then the classification module of the *SVM^{light}* software. The classification results are then calculated for the three users.

Table 8 shows the masquerade detection results for each of the three users A, B, and C when only the mouse features are considered for training and classification. It should be noted that the high classification rate for user B is balanced by the fact that the estimation error for this classification is $\approx 20\%$, which may bring down the lowest correct classification rates to $\approx 80\%$. It is interesting to note that for user C

Table 8

Classification rates for users A, B, and C with 740 training and 360 testing samples for 16 mouse features; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	97.22%/ 350 (KC)	2.78%/ 10 (IC)	135	≤18.11
B → (A, C)	100%/ 360 (KC)	0.00%/ 0 (IC)	152	≤20.14
C → (A, B)	88.61%/ 319 (KC)	11.39%/ 41 (IC)	475	≤63.65

Table 9

Classification rates for users A, B, and C with 740 training and 360 testing samples for 16 mouse features and single keystroke time; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	87.78%/ 316 (KC)	12.22%/ 44 (IC)	112	≤14.19
B → (A, C)	100%/ 360 (KC)	0.00%/ 0 (IC)	158	≤21.08
C → (A, B)	78.89%/ 284 (KC)	21.11%/ 76 (IC)	472	≤62.84

the number of support vectors is high, making it difficult for the classifier to make a decision, thus a high estimation error.

Table 9 shows the classification rates for the mouse as well as single keystroke time features for these three users. The correct classification rates and estimation error for user B compensate each other in this table also. User C has high estimation error in this case as well.

Table 10 shows the classification rates for the three users when mouse and di-graph latency features are considered. Table 11 shows the classification rates for the three users when mouse, single keystroke time, and di-graph latency features are considered. Table 12 shows the classification rates for the three users when mouse and tri-graph latency features are considered.

It is useful to observe the correct classification rates for these users and analyze the factors affecting the rates. The correct classification rates for users A, B, and C are plotted against the following five features sets:

- Mouse features only.
- Mouse features and single character keystroke time.
- Mouse features and di-graph latencies.
- Mouse features, single character keystroke time, and di-graph latencies.
- Mouse features and tri-graph latencies.

Figure 9 shows the correct classification rate variations for user A when tested against users B and C. The best rates are achieved when mouse features are combined with the di-graph latencies. This leads to the conclusion that for user A, the most *unique* features are based on mouse movements and the di-graphs. The other interesting thing to note is that the single keystroke time values for user A are not

Table 10

Classification rates for users A, B, and C with 740 training and 360 testing samples for 16 mouse features and di-graph latencies; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	92.50%/ 333 (KC)	7.50%/ 27 (IC)	90	≤11.08
B → (A, C)	97.22%/ 350 (KC)	2.78%/ 10 (IC)	129	≤14.05
C → (A, B)	69.44%/ 250 (KC)	30.56%/ 110 (IC)	236	≤29.46

Table 11

Classification rates for users A, B, and C with 740 training and 360 testing samples for 16 mouse features, single keystroke time and di-graph latencies; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	89.44%/ 322 (KC)	10.56%/ 38 (IC)	97	≤11.08
B → (A, C)	97.22%/ 350 (KC)	2.78%/ 10 (IC)	128	≤15.27
C → (A, B)	69.44%/ 250 (KC)	30.56%/ 110 (IC)	249	≤30.14

Table 12

Classification rates for users A, B, and C with 740 training and 360 testing samples for 16 mouse features and tri-graph latencies; KC = correctly classified, IC = incorrectly classified

Model	Correct classification rate	False alerts	Support vectors	Estimation error (%)
A → (B, C)	91.67%/ 330 (KC)	8.33%/ 30 (IC)	93	≤10.27
B → (A, C)	100%/ 360 (KC)	0.00%/ 0 (IC)	126	≤13.38
C → (A, B)	77.78%/ 280 (KC)	22.22%/ 80 (IC)	202	≤21.49

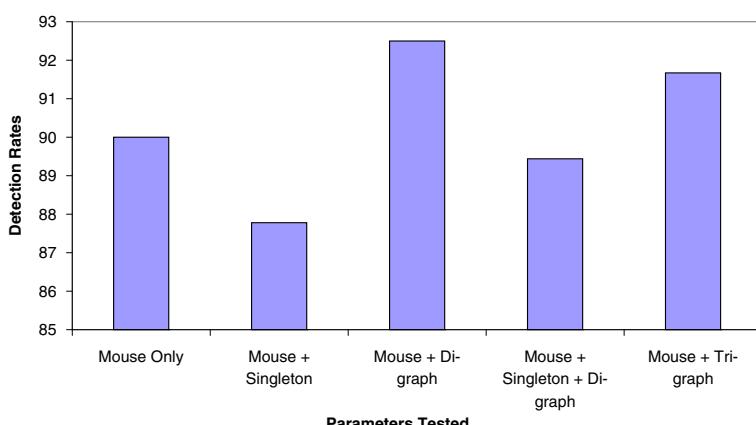


Fig. 9. Classification rates for user A with training data from users B and C.

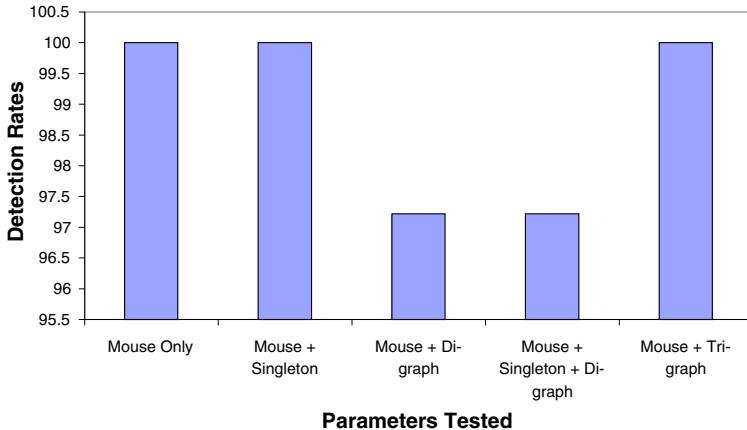


Fig. 10. Classification rates for user B with training data from users A and C.

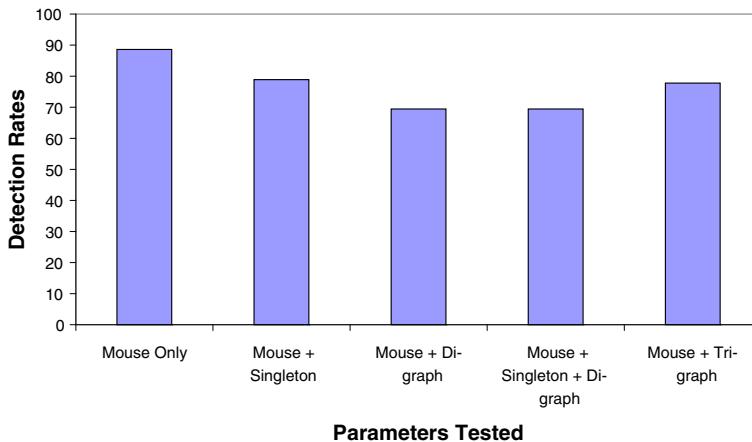


Fig. 11. Classification rates for user C with training data from users A and B.

very unique from other users and thus lead to the reduced correct classification rates when combined with mouse features.

Figure 10 shows the correct classification rate variations for user B when tested against users A and C. The rates suffer due to the inclusion of di-graph latencies, which indicate the lack of enough data for proper calculation of these latencies or the fact that these latencies do not provide enough *uniqueness* for the classification.

Figure 11 shows the correct classification rate variations for user C when tested against users A and B. The mouse data alone for user C provides the better classification rate. This indicates that user C has *unique* mouse characteristics from other users. Also, the di-graph latencies decrease the correct classification rate, allowing for the fact that these values are not unique among the users.

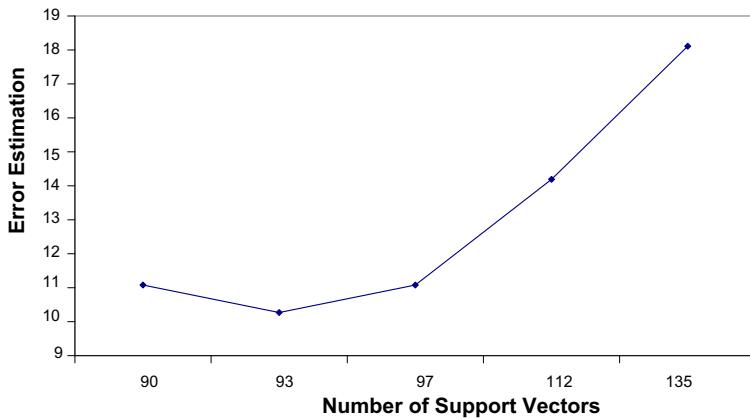


Fig. 12. User A: estimation error vs. support vectors generated.

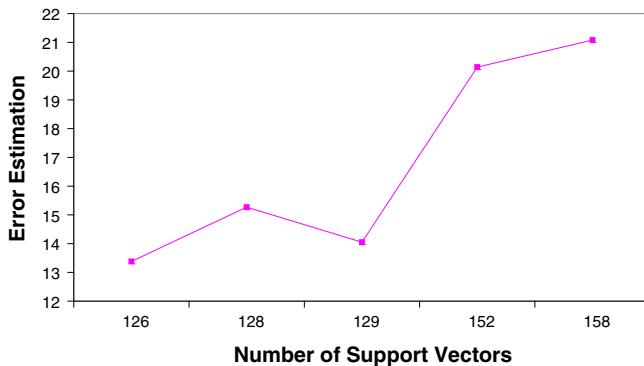


Fig. 13. User B: estimation error vs. support vectors generated.

It is also interesting to note that the number of support vectors (i.e., the training points nearest to the hyperplane) play an important role in determining the estimation error for these classification results. Figure 12 shows the estimation error increase for user A with respect to the increase in the number of support vectors.

Figure 13 shows the estimation error increase for user B with respect to the increase in the number of support vectors. It should be noted that although the estimation error increases with the increase in support vectors, there are instances when the support vectors are strategically placed near the hyperplane and thus do not affect the correct classification rate adversely. This can be seen in Fig. 13.

Figure 14 shows the estimation error increase for user C with respect to the increase in the number of support vectors. This figure also shows the trend of increase in estimation error with the increase in number of support vectors.

Figure 15 shows the trend in improving the correct classification rate with increase in the supplied data for training and classification. This trend strengthens the belief

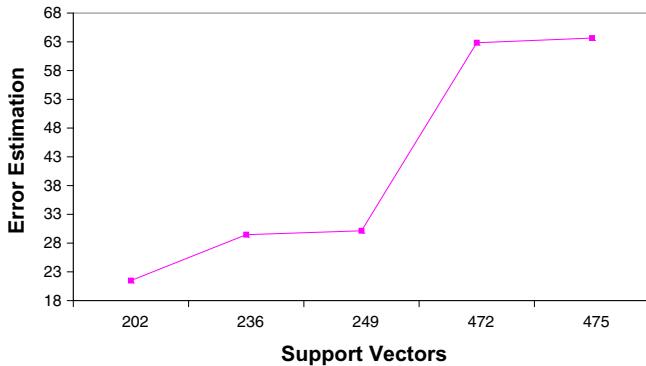


Fig. 14. User C: estimation error vs. support vectors generated.

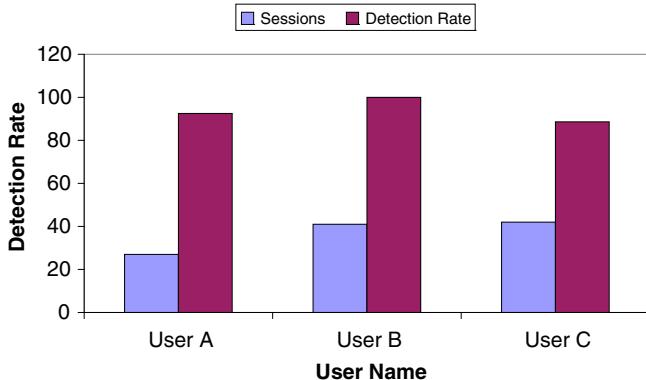


Fig. 15. Correct classification rate variation based on total number of sessions used.

that more training data could improve the classification accuracy of the IDSs and reduce the estimation error.

Although the results presented are based on a set of three users, it is possible to construct feature vectors for more users and test the system for its accuracy. However, it is interesting to note that the 16 features presented in Section 5.1 provide good accuracy and increasing the features may only marginally be beneficial based on the type of features used and the uniqueness of these features.

It should be noted that the sample size for the experiment was limited by the resource in terms of how many users were available and willing to get their profiles being used for research. Although, more number of users would ensure that the experiment is scalable and could be deployed in a larger scenario, the quantity of data for each user being profiled played a more important role than the actual number of users for our analysis due to the significance of the specific features. Since the objective of our experiments was to determine the feasibility of the approach, the small number of users is not really a limitation as long as enough data is collected

from these users. The scalability part of the experiment, on the other hand, would require a large set of users and was outside the scope of our research.

5.3. Classification rate issues

It is important to observe that the classification results presented here are influenced by the paucity of data and the fact that the users are aware of the logging mechanism for building profiles. Once these factors are overcome, the rates may vary and provide more accurate results associated with various users.

Other factors which may affect the accuracy of presented profiling tool are users' physical fitness level, their mood during the time logging is done and users' skill for performing the job function. Although these factors will affect some of the features related to user behavior profiles, it is assumed that *unique* features would not be affected and hence will not affect the overall classification rates significantly.

6. Discussion and conclusion

This chapter has presented a new empirically driven framework for constructing comprehensive feature vectors for GUI-based operating systems, which has been shown to offer good masquerade detection accuracy. After utilizing the methodology to collect data and extract and select features as described in Section 4, feature vectors have been constructed from these features and support vector machine algorithms have been used to first train and then classify the users. It has been found that user behavior features based on the mouse activity and keystroke dynamics on a GUI-based system can be used to uniquely identify users and thus provide better masquerade detection capability in GUI-based systems.

The profile generation tool has been tested with a limited set of data. This is due to the fact that there are no public datasets available for GUI-based systems. Data from only three users was available at the time of experimentation, so it will benefit to test the presented system with data from more users. Also, it has been found that the mouse-related 16 features could provide good classification accuracy using SVM with the current set of users and increasing this feature set could only provide marginal improvements.

Bhukya et al. (2007) extended the approach of masquerade detection using GUI data, from our preliminary work reported in Garg et al. (2006), for the K Desktop Environment (KDE) in Linux systems. Their results are promising and provide documented proof that our overall approach of using GUI data for masquerade detection is scalable and can provide good results in a large and heterogeneous environment.

Acknowledgments

The authors thank S. Vidyaraman and Ragini Rahalkar for their help in implementation. This research was supported in part by U.S. Air Force Research Lab, under contract number: F30602-00-10507.

References

- Ahmed, A.A.E., Traore, I., 2005. Anomaly intrusion detection based on biometrics. In: Proceedings of the IEEE Workshop on Information Assurance 2005, West Point, NY, USA, June 15–17.
- Brown, D., Claypool, M., 2003. Curious browsers: automated gathering of implicit interest indicators by an instrumented browser. In: Proceedings of Workshop on Implicit Measures of User Interests and Preferences. Worcester Polytechnic Institute.
- Brank, J., Grobelnik, M., Milić-Frayling, N., Mladenović, D., 2002. Feature Selection Using Linear Support Vector Machines. Technical Report MSR-TR-2002-63, Microsoft Research, One Microsoft Way, Redmond, WA 98052, 12 June.
- Bergadano, F., Gunetti, D., Picardi, C., 2002. User authentication through keystroke dynamics. *ACM Trans. Inform. Syst. Secur.* 5, 367–397.
- Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152.
- Bhukya, W., Kommuru, S., Negi, A., 2007. Masquerade detection based upon GUI user profiling in Linux systems. *Lect. Notes Comput. Sci.* 4846, 228–239.
- Claypool, M., Le, P., Waseda, M., Brown, D., 2001. Implicit interest indicators. In: Proceedings of ACM Intelligent User Interfaces Conference (IUI), Santa Fe, New Mexico, USA, January 14–17.
- Cortes, C., Vapnik, V.N., 1995. Support-vector networks. *Mach. Learn.* J. 20, 273–297.
- Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification, second ed. John Wiley and Sons Inc.
- Denning, D.E., Neumann, P.G., 1985. Requirements and Model for IDES—A Real-time Intrusion Detection System. Technical Report, Computer Science Laboratory, SRI International, Menlo Park, CA.
- Forrest, S., Hofmeyr, S.A., Somayaji, A., 1997. Computer immunology. *Commun. ACM* 40 (10), 88–96.
- Feng, H., Kolesnikov, O., Fogla, P., Lee, W., Gong, W., 2003. Anomaly detection using call stack information. In: Proceedings of IEEE Symposium on Security and Privacy, Oakland, California, May.
- Gupta, A., Asthana, A., Gupta, N., 2008. Masquerade detection using typing pattern. In: Proceedings of Second National Conference on Challenges and Opportunities in Information Technology (COIT-2008), Mandi Gobindgarh, Punjab, India, March 29.
- Giot, R., Hemery, B., Rosenberger, C., 2010. Low cost and usable multimodal biometric system based on keystroke dynamics and 2d face recognition. In: 20th International Conference on Pattern Recognition 2010 (ICPR), pp. 1128–1131.
- Gunetti, D., Picardi, C., 2005. Keystroke analysis of free text. *ACM Trans. Inform. Syst. Secur. (ACM TISSEC)* 8(3), 312–347.
- Garg, A., Rahalkar, R., Upadhyaya, S., Kwiat, K., 2006. Profiling users in GUI based systems for masquerade detection. In: Proceedings of 7th Annual IEEE Information Assurance Workshop (IAW 2006), United States Military Academy, West Point, New York, June 21–23.
- Goecks, J., Shavlik, J., 1999. Automatically labeling web pages based on normal user actions. In: IJCAI Workshop on Machine Learning for Information Filtering.
- Ghosh, A., Schwartzbard, A., Schatz, M., 1999. Learning program behavior profiles for intrusion detection. In: First USENIX Workshop on Intrusion Detection and Network Monitoring, pp. 51–62.
- Hofmeyr, S., Forrest, S., Somayaji, A., 1998. Intrusion detection using sequences of system calls. *J. Comput. Secur.* 6 (3), 151–180.
- Imsand, E.S., Hamilton, J.A., 2007. GUI usage analysis for masquerade detection. In: Proceedings of the Information Assurance and Security Workshop, June 20–22, pp. 270–276.
- Ilgun, K., Kemmerer, R., Porras, P., 1995. State transition analysis: a rule-based intrusion detection approach. *Softw. Eng.* 21 (3), 181–199.
- Imsand, E., 2008. Applications of GUI usage analysis. PhD Thesis, Auburn University.
- Joachims, T., 1998. Text categorization with support vector machines: learning with many relevant features. In: Proceedings of 10th European Conference on Machine Learning ECML-98. Springer-Verlag, Heidelberg, DE, pp. 137–142.
- Joachims, T., 2001. A statistical learning model of text classification for support vector machines. In: The 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, September 9–12.

- Joachims, T., 2002. Learning to Classify Text Using Support Vector Machines: Methods. Kluwer Academic Publishers, Theory and Algorithms.
- Joachims, T., 2004. SVM Light: Support Vector Machine. Department of Computer Science, Cornell University. <http://www.cs.cornell.edu/People/tj/svm_light/index.html>.
- Javitz, H.S. and Valdes, A., 1991. The SRI IDES Statistical Anomaly Detector. In: Proceedings of the IEEE Research in Security and Privacy, Oakland, CA, pp. 316–376.
- Lane, T., 1999. Hidden Markov Models for human-computer interface modeling. In: Proceedings of IJCAI-99 Workshop on Learning About Users, pp. 35–44.
- Lane, T., Brodley, C., 1997a. Sequence matching and learning in anomaly detection for computer security. In: Proceedings of AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management, pp. 43–49.
- Lane, T., Brodley, C.E., 1997b. An application of machine learning to anomaly detection. In: Proceedings of Twentieth National Information Systems Security Conference, Gaithersburgh, MD, vol. 1, pp. 366–380 (The National Institute of Standards and Technology and the National Computer Security Center).
- Lane, T., Brodley, C.E., 1999. Temporal sequence learning and data reduction for anomaly detection. ACM Trans. Inform. Syst. Secur. 2 (3), 295–331.
- Levitt, K., Ko, C., Fink, G., 1994. Automated detection of vulnerabilities in privileged programs by execution monitoring. In: Computer Security Application Conference.
- Lee, W., Stolfo, S., Mok, K., 1999. A data mining framework for building intrusion detection models. In: IEEE Symposium on Security and Privacy, pp. 120–132.
- Li, Y., Wu, N., Jajodia, S., Wang, S., 2002. Enhancing profiles for anomaly detection using time granularities. J. Comput. Secur. 10 (1,2), 137–157.
- Marceau, C., 2000. Characterizing the behavior of a program using multiple-length N-grams. In: Proceedings of the Workshop on New Security Paradigms 2000, Ballycotton, County Cork, Ireland, pp. 101–110.
- Maxion, R.A., 2003. Masquerade detection using enriched command lines. In: Proceedings of International Conference on Dependable Systems and Networks (DSN 2003), San Francisco, CA.
- Michael, C., Ghosh, A., 2000. Using finite automata to mine execution data for intrusion detection: a preliminary report. Lect. Notes Comput. Sci. 1907.
- Mueller, F., Lockerd, A., 2001. Cheese: tracking mouse movement activity on websites, a Tool for user modeling. In: Proceedings of ACM SIGCHI Conference on Computer-Human Interaction (CHI).
- Monrose, F., Rubin, A., 1997. Authentication via keystroke dynamics. In: ACM Conference on Computer and Communications, Security, pp. 48–56.
- Maxion, R.A., Townsend, T.N., 2002. Masquerade detection using truncated command lines. In: Proceedings of International Conference on Dependable Systems and Networks (DSN 2002), pp. 219–228.
- Maxion, R.A., Townsend, T.N., 2004. Masquerade detection augmented with error detection. IEEE Trans. Reliability, Special Sec. Qual./Reliability Eng. Inform. Syst. 53 (1), 124–147.
- Orr, M.J.L., 1996. Introduction to Radial Basis Function Networks, Institute for Adaptive and Neural Computation, Edinburgh Univ.
- Pusara, M., Brodley, C.E., 2004. User re-authentication via mouse movements. In: Proceedings of the ACM Workshop on Visualization and Data Mining for Computer Security 2004 VizSEC/DMSEC 2004, Washington, DC, USA, pp. 1–8.
- Porras, P.A., Neumann, P.G., 1997. EMERALD: event monitoring enabling responses to anomalous live disturbances. In: Proceedings of 20th NIST-NCSC National Information Systems Security Conference, pp. 353–365.
- Rajagopalan, M., Debray, S., Hiltunen, M., Schlichting, R., 2002. Profile-directed Optimization of Event-based Programs. In: Proceedings of the ACM SIGPLAN 2002 Conference on Programming Language Design and Implementation (PLDI), pp. 106–116.
- Richardson, R., 2008. CSI Computer Crime and Security Survey. Computer Security Institute.
- Revett, K., Jahankhani, H., Magalhaes, S.T., Santos, H.M.D., 2008. A survey of user authentication based on mouse dynamics. In: Global E-Security, Communications in Computer and Information Science. Springer, pp. 210–219.
- Schonlau, M., 1998. Masquerading User Data. <<http://www.schonlau.net/intrusion.html>>.

- Schonlau, M., DuMouchel, W., Ju, W., Karr, A., Theus, M., Vardi, Y., 2001a. Computer intrusion: detecting masquerades. *Stat. Sci.* 16 (1), 58–74.
- Schonlau, M., DuMouchel, W., Ju, W.-H., Karr, A.F., Theus, M., Vardi, Y., 2001b. Computer intrusion: detecting masquerades. *Stat. Sci.* 16, 58–74.
- Shavlik, J., Shavlik, M., Fahland, M., 2001. Evaluating software sensors for actively profiling Windows 2000 computer users. In: Fourth International Symposium on Recent Advances in Intrusion Detection.
- Vapnik, V., 1982. Estimation of Dependencies Based on Empirical Data. Springer.
- Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer.
- Vapnik, V., 1998. Statistical Learning Theory. Wiley, Chichester, GB.
- Wagner, D., Dean, D., 2001. Intrusion Detection via Static Analysis. In: IEEE Symposium on Security and Privacy, pp. 156–169.
- Wespi, A., Dacier, M., Debar, H., 2000. Intrusion detection using variable-length audit trail patterns. In: Recent Advances in Intrusion Detection—Lecture Notes in Computer Science, vol. 1907. Springer-Verlag, pp. 110–29.
- Warrender, C., Forrest, S., Pearlmuter, B., 1999. Detecting intrusions using system calls: alternative data models. In: IEEE Symposium on Security and Privacy, Oakland, CA, pp. 133–145.
- Wikipedia, 2012. Support Vector Machine. <http://en.wikipedia.org/wiki/support_vector_machines>.
- Wang, K., Stolfo, S.J., 2003. One class training for masquerade detection. In: ICDM Workshop on Data Mining for Computer Security (DMSEC 2003).
- Ye, N., 2000. A Markov chain model of temporal behavior for anomaly detection. In: Proceedings of the IEEE Systems, Man, and Cybernetics Information Assurance and Security, Workshop 2000, pp. 171–174.

This page is intentionally left blank

Application of Bayesian Graphical Models to Iris Recognition

B.V.K. Vijaya Kumar¹, Vishnu Naresh Boddehi¹, Jonathon M. Smereka¹, Jason Thornton², and Marios Savvides¹

¹*Electrical and Computer Engineering Carnegie Mellon University,
Pittsburgh, PA, USA*

²*Lincoln Laboratory, Lexington, MA, USA*

Abstract

Recognition of humans based on their biometric signatures is becoming of increasing importance because of its applications in access security, reliable identification for benefits distribution and homeland security among others. Popular biometric modalities include face images, fingerprints, iris images, palm prints, gait patterns, voice, etc. Iris images are of significant interest because of the excellent recognition rates they offer in controlled image acquisition conditions where the image quality is expected to be good enough to capture the details of the iris. However, in more realistic scenarios, iris images may not be of necessary quality because of image distortions and occlusions due to eyelids and eyelashes and the recognition rates provided by standard approaches may not be sufficient. In this chapter, we discuss how Bayesian graphical models can be used to achieve improved iris recognition in the presence of image impairments such as nonlinear deformations and occlusions. We illustrate the performance of these methods on sample iris image data sets.

Keywords: iris, biometrics, graphical models, correlation filters, FOCS data, ROC curve, MAP estimation

1. Introduction

Because of increasing security requirements, it is important to be able to identify an individual reliably. Conventional human identification methods rely on passwords and/or possession of physical items such as RFID devices or keys. The problem with these approaches is that passwords can be forgotten and keys and tokens can be lost or stolen. It is preferred that a person's identity is based on who he/she is rather

than what they know or what they possess. Biometric signatures offer an excellent opportunity to connect a person's identity to that person.

A biometric is a physical or behavioral signature of a person which allows a computer system to distinguish that individual from others. Examples of biometrics include face images, fingerprints, iris images, palm prints, gait patterns, ear images, and others. Biometric matching involves matching the biometric signature (e.g., fingerprints) of a person acquired during enrollment (also called training) stage against the signatures collected during verification (also called testing) stage. In some applications (e.g., in systems designed to verify the claimed identity of a person to authorize access privileges for that person), an enrolled biometric signature (called a gallery image) is compared against a test biometric signature (called a probe image) in a 1:1 comparison. In other applications (e.g., when needing to determine whether a person requesting benefits has previously received such benefits and is in the corresponding database), we have to match the test biometric against several enrolled biometric signatures in a 1:N comparison. In both settings, it is desired that the match score be high when matching pairs of biometric signatures from the same person and the match scores be low for pairs of signatures from different individuals.

One biometric modality receiving much attention is the iris pattern, which is the texture-rich region of the eye bordered by the black pupil inside and the white sclera outside. Figure 1 shows a sample image of the iris.

The iris pattern is believed to be unique and is also believed to be more or less stable over one's life time (Adler, 1965). It has been observed that the left and right irises from the same person are distinct and identical twins have different iris patterns. Because of these features, iris has become one of the major biometric modalities being investigated for human identification.

Some biometric modalities such as fingerprints require user effort whereas other biometric modalities such as face images and gait patterns may be acquired without requiring user effort. Early iris image acquisition systems required the user to place their eye close to the camera whereas some of the more recent stand-off iris image acquisition systems (Matey et al., 2006) can acquire the user's iris from an increased distance. For iris image acquisition, illumination is usually in near-infrared (NIR) wavelengths to exhibit better contrast of the iris to the pupil and sclera. Further, strong visible light can cause pupil dilation which changes the geometric properties of the iris region, making it more difficult to match iris images of the same eye acquired under different light levels. Use of NIR illumination alleviates the pupil dilation problem to some extent since humans do not react to NIR illumination as they do to visible light. For safety reasons, the NIR illumination levels cannot be high which in turn limits the available distance between the camera and the eye during acquisition.

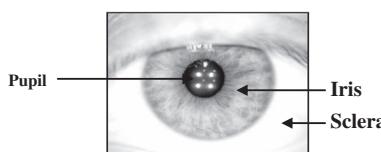


Fig. 1. Iris pattern, contained in an eye image.

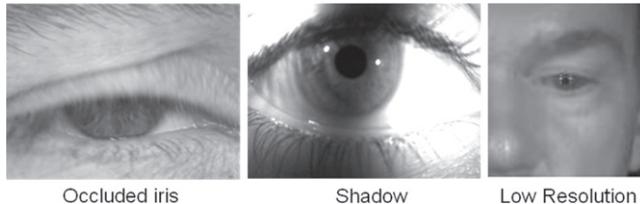


Fig. 2. Example iris images with impairments.

Many methods have been proposed for iris image matching. Wildes (1997) decomposed the iris signal into different frequency bands, using normalized correlation with templates to perform recognition. Other feature extraction techniques have involved projecting the iris pattern onto an independent components analysis (ICA) basis (Huang et al., 2002) or a Fractional Fourier basis (Yu et al., 2002). Also, iris patterns have been characterized by their location in global principal component analysis (PCA) subspaces (Dorairaj et al., 2005).

The most well-known technique for iris matching uses Gabor wavelet analysis introduced by Daugman (1993). Gabor wavelet coefficients are good at representing the texture in the iris images, since they represent different local spatial frequencies, orientation, and bandwidths. The complex values resulting from Gabor analysis are quantized to obtain a binary code representing that iris. The Hamming distance between the iris codes of the enrollment iris image and verification images are then used to determine the degree of match between the two iris images. Authentic pairs should exhibit small Hamming distances whereas impostor pairs should have large Hamming distances. Although the iris code gives excellent results when iris images are acquired in controlled conditions, the recognition rates deteriorate in less controlled conditions where the images exhibit impairments such as deformations, occlusions, specular reflections, and geometrical changes from gaze differences. Figure 2 shows some examples of iris images with such impairments. In this chapter, we will discuss how Bayesian graphical models can be used to achieve improved performance when matching such iris images.

The rest of this chapter is organized as follows. In Section 2, we discuss the main components of Gabor wavelet algorithm in more detail. This is followed by a discussion in Section 3 of an alternative iris matching approach that is based on correlation filters. Section 4 discusses how Bayesian graphical models can be used to achieve improved iris matching and Section 5 provides a summary.

2. Gabor wavelet-based matching

The Gabor wavelet-based matching algorithm consists of following main components:

- *Iris segmentation*: This is used to isolate the texture-rich iris portion of the image from the rest of the image.
- *Iris unwrapping*: This is used to convert the image from the Cartesian coordinate system to a polar coordinate system.

- *Gabor wavelet encoding*: In this step, various Gabor wavelets are applied to the unwrapped iris image and the resulting complex outputs are quantized to produce a binary code
- *Hamming distance computation*: During matching stage, Hamming distance binary codes from two iris images are computed and normalized.

In the following subsections, we will briefly review each of the above major steps.

2.1. Iris segmentation

The first step in iris matching is segmenting the texture-rich iris portion from the eye image that contains other parts such as the pupil and the sclera. Let $I(x, y)$ represent the eye image in Cartesian coordinates. Integrating this intensity image on a circle of radius r centered at x_c, y_c leads to the following output:

$$p(x_c, y_c, r) = \int_0^{2\pi} I(x_c + r \cos \theta, y_c + r \sin \theta) d\theta. \quad (1)$$

In practice, we may exclude integration over regions that are occluded by eye lids and other obstructions. Since the pupil region is darker than the iris region and the sclera region is brighter than the iris region, the inner and outer boundaries are found by determining the values x_c, y_c , and r that maximize the magnitude of $\left[\frac{\partial p(x_c, y_c, r)}{\partial r} \right]$. An example of iris segmentation is shown in Fig. 3.

2.2. Iris unwrapping

Once the iris boundaries are obtained, next step is to convert the iris from Cartesian coordinates to polar coordinates as shown in Fig. 4. Here the iris within the found inner and outer boundaries is divided into a polar grid and iris intensity values at the polar grid intersections are mapped into the rectangular “unwrapped” format shown on the right side of Fig. 4. Within the example, the horizontal axis denotes the angle and the vertical axis denotes radial distance from the inner boundary. The mapping is normalized so that the inner boundary maps to $\rho = 0$ in the unwrapped image and outer boundary maps to $\rho = 1$ in the unwrapped image.

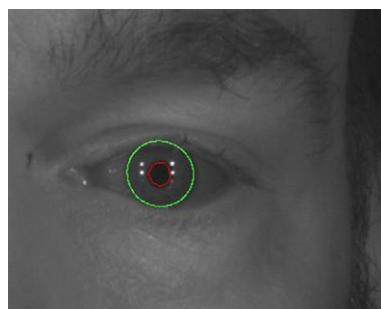


Fig. 3. Example iris segmentation.

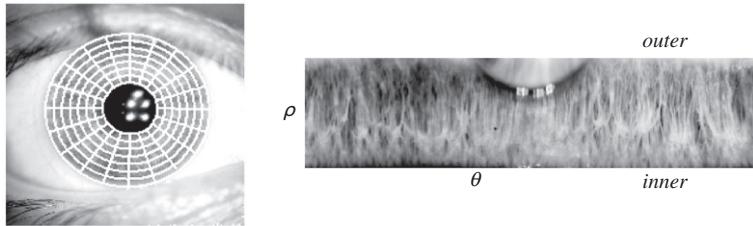


Fig. 4. Iris unwrapping.

Segmentation and unwrapping enable us to map every iris pattern into the same rectangular area in the polar coordinate system. A benefit of this unwrapping is that it normalizes for translation and scale of the iris pattern, as well as pupil dilation. However, in-plane rotations of the iris image result in cyclic shifts along θ .

2.3. Iris encoding

In this subsection we discuss iris encoding using Gabor wavelet analysis. Several variants of this have been proposed in the literature, but we will describe the most basic version of this method. A 2D Gabor wavelet has the functional form of a complex exponential with a Gaussian envelope. This is given in polar coordinates as

$$\psi(\rho, \theta) = \exp \left[-\frac{\rho^2}{2\sigma_\rho^2} - \frac{\theta^2}{2\sigma_\theta^2} - j\omega_\theta \theta - j\omega_\rho \rho \right], \quad (2)$$

where σ_ρ and σ_θ denote the widths in the radial and angular directions, respectively, and ω_ρ and ω_θ denote the radial and angular frequencies, respectively. The Gaussian envelope localizes the wavelet in space, while the complex exponential localizes the wavelet in frequency. Figure 5 shows the real part and the imaginary part of an example Gabor wavelet.

Choosing different parameter values for Gabor wavelets and translating them by different amounts results in a family of (not necessarily orthogonal) Gabor wavelets. The unwrapped iris image is projected onto the Gabor wavelet basis to yield a set of complex coefficients. As shown in Fig. 6, depending on which quadrant each

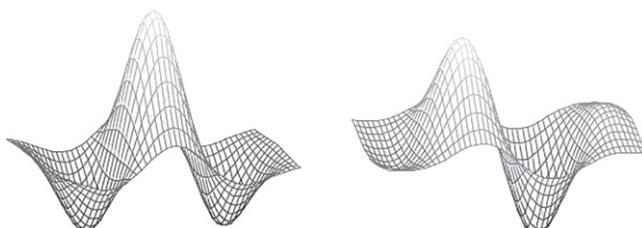


Fig. 5. Real part (left) and imaginary part (right) of a Gabor wavelet.



Fig. 6. Encoding of the complex coefficient by two bits.

complex coefficient falls into, a corresponding bit pattern is encoded as the result. When the Gabor wavelet family produces N complex coefficients, we get a binary code with $2N$ bits. A match score between two iris patterns is then the fraction of matching bits between their characteristic codes.

A publicly available implementation for generating the binary code (Masek and Kovesi, 2003) uses a basis of 1D log-Gabor wavelets. This basis (which produces a code on the order of 10^4 – 10^5 bits) has only one orientation (horizontal) but many translations.

2.4. Iris matching

Once iris patterns are represented by binary codes, the similarity between two iris patterns can be represented by the Hamming distance (i.e., the number of bits that are different) between their binary codes. By dividing this Hamming distance by the number of bits in the binary code, we obtain the normalized Hamming distance (NHD) between the two iris patterns. Ideally, NHD for iris patterns from the same eye should be zero and NHD for iris patterns from different eyes should be close to 1/2 since two random binary patterns will agree in half the bits and disagree in the remaining half.

In reality, the intra-class variation of iris patterns causes variability in corresponding binary codes leading to a spread of NHD values for the impostor pairs and authentic pairs. Any overlap between these NHD distributions results in recognition errors. These errors are known as either a false accept (i.e., binary codes for two different eyes have NHD smaller than a chosen threshold—falsely accepting the impostor) or a false reject (i.e., binary codes from same eye have NHD larger than the threshold—falsely rejecting the authentic). When the iris images are of high quality, excellent recognition rates have been demonstrated using the binary code approach.

3. Correlation filter-based iris matching

Another approach to iris matching is the use of correlation filters (Vijaya Kumar et al., 2005). A correlation filter (the phrase “filter” is used to indicate that the design is in the spatial frequency domain rather than image domain) is designed specifically for the recognition of one pattern class. When a test or probe image is presented, it is segmented and unwrapped and then cross-correlated with the template represented by the correlation filter to produce a correlation output.

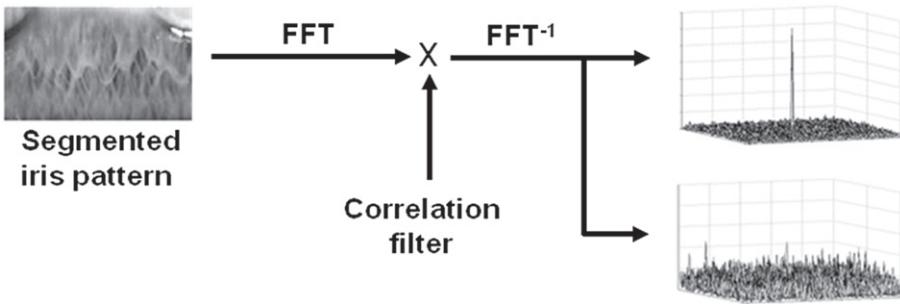


Fig. 7. Cross-correlation of a segmented and unwrapped iris pattern.

The cross-correlation between two images is efficiently computed using 2-D fast Fourier transform (FFT) as shown schematically in Fig. 7. The resulting correlation output should exhibit a sharp peak if there exists a match between the template and the probe image. If the template and the probe correspond to different eyes, the correlation output should not exhibit a dominant peak.

One way to quantify the degree of similarity between the probe and template is the correlation peak value. However, this peak value can be affected by the illumination level of the probe image. Achieving some tolerance to illumination levels can be done by using peak-to-sidelobe ratio (PSR), defined here as $\text{PSR} = (\text{peak} - \mu)/\sigma$ where peak denotes the peak value and where μ and σ denote the mean and the standard deviation, respectively, of the correlation output. Correlation filters are shift-invariant, i.e., if the probe image is translated by some amount, then the correlation output will be shifted by the same amount. Since the probe image is in polar coordinates, this shift-invariance provides us with the ability to tolerate iris image rotations in the Cartesian coordinate system.

Over the last twenty years, many advanced correlation filter designs have been introduced, particularly for automatic recognition applications (Vijaya Kumar, 1992). In the early correlation filter designs known as *synthetic discriminant function* (SDF) filters, the template is designed so that the correlation peaks (resulting when centered training images are cross-correlated with this template) take on pre-specified values. For example, authentic training images result in a correlation peak at the origin of $C(0, 0) = 1$, while all imposter training images lead to $C(0, 0) = 0$. The expectation is that such a template will yield values close to 1 at the correlation output origin in response to centered authentic images and will lead to small values in response to impostor images.

Since the probe image may not be perfectly centered, it is not easy to know where the correlation output's center is supposed to be. To overcome this problem, minimum average correlation energy (MACE) filters were introduced (Mahalanobis et al., 1987) where the template is designed to satisfy the above correlation peak constraints while simultaneously minimizing average correlation energy (ACE). ACE is the average of the energies in the correlation outputs for all training images and minimizing ACE while satisfying the peak constraint of 1 for authentic images will result in a very visible peak for the correlation output for authentic images as shown in Fig. 7. This makes it easy to determine that the input is from the authentic

class even if the image is not centered. For impostor images, the output will not have a noticeable peak.

Although the MACE filter can produce sharp peaks in response to authentic training images, it suffers from the problem that is overly sensitive to noise in the input image. This is mainly because MACE filters end up amplifying the high-frequency content in the input image (in order to produce a sharp correlation peak) and in return end up amplifying high-frequency portions of the noise in the input image. One way to reduce this noise sensitivity is to reduce the output noise variance (ONV) which measures the variance of the output in response to input noise.

Minimizing ONV and minimizing ACE are conflicting as minimizing ACE leads to filters that emphasize high frequencies whereas minimizing ONV leads to filters that emphasize low frequencies. One way to tradeoff ACE for ONV is to use the Optimal Trade-off Synthetic Discriminant Function (OTSDF) filter (Refregier, 1990). Let \mathbf{h} be a vector containing the scanned versions of the correlation filter (in frequency domain) and let \mathbf{D} represent a diagonal matrix with the average power spectrum of the training images; then the ACE can be written as

$$ACE = \mathbf{h}^+ \mathbf{D} \mathbf{h}, \quad (3)$$

where $+$ denotes conjugate transpose. The ONV characterizes the variance of the noise at the correlation output. If the input noise is white, ONV is given by

$$ONV = \mathbf{h}^+ \mathbf{h}. \quad (4)$$

Since minimizing ACE and ONV are conflicting goals, OTSDF approach minimizes one (e.g., ACE) while holding the other (namely, ONV) below a given value. It has been shown that, because ACE and ONV are quadratic functions of \mathbf{h} , this multi-criteria minimization can be solved by minimizing a weighted sum of ACE and ONV, subject to the correlation peak constraints. Resulting OTSDF filter is given as follows:

$$\mathbf{h} = \mathbf{A}^{-1} \mathbf{X} (\mathbf{X}^+ \mathbf{A}^{-1} \mathbf{X})^{-1} \mathbf{u}, \quad (5)$$

with $\mathbf{A} = \alpha \mathbf{I} + \sqrt{1 - \alpha^2} \mathbf{D}$. Vector \mathbf{u} contains the correlation peak constraints (1 for authentics, 0 for imposters), and the i th column of matrix \mathbf{X} contains the vectorized 2D-Fourier transform of the i th training image denoted as \mathbf{x}_i .

Over the past two decades, correlation filter designs have evolved significantly. The advances include relaxing the correlation peak constraints, designs that consider the entire correlation plane and not just the correlation peak, correlation filters that exhibit tolerance to in-plane rotations and scale changes, nonlinear versions such as the quadratic correlation filters, and multi-frame correlation filters that can be applied to video sequences, not just images.

To illustrate the recognition performance of correlation filters, we will describe the results on a database of visible light iris images collected at Carnegie Mellon University (CMU). The database contains normal within-class iris image variations including rotation, scale change, pupil dilation, eyelid occlusion, and focus variability. In addition, iris images do not provide very good contrast in visible wavelengths making the recognition of these images even more challenging.

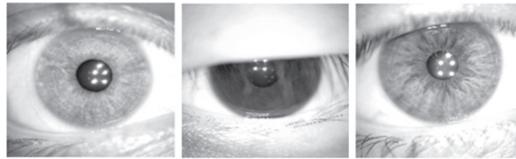


Fig. 8. Sample iris images from CMU database.

The CMU iris database was collected under visible light illumination and contains 101 iris classes. Each class consists of approximately 20–25 images from the same eye, collected in two different sessions up to six months apart. The original images have much higher resolution than needed (approximately 11 megapixels); consequently, they are downsampled before processing. Figure 8 shows sample images.

We compare the performance of advanced correlation filter designs to the Gabor wavelet encoding methods (both the original Gabor wavelet method as well as the 1D log Gabor wavelet method) described in Section 2. We extracted sections of the iris not likely to experience eyelid occlusion and applied all the classifiers to the unoccluded sections.

We used five images randomly chosen from each iris class as training data. The remaining 20 images per class were used for the testing. In total, this gave approximately 2000 authentic comparisons and 200,000 imposter comparisons. On the CMU database, we applied OTSDF correlation filter classification and both implementations of Gabor wavelet encoding classification. The match scores are compared to a threshold to determine whether the probe and template correspond to the same eye or different eyes. False accept rate (FAR) refers to the fraction of impostor pairs that were incorrectly labeled as belonging to the same eye and false reject rate (FRR) refers to the fraction of authentic pairs that were incorrectly determined as belonging to different eyes. By varying the threshold, FAR can be traded off for FRR and resulting receiver operating characteristic (ROC) curve is shown in Fig. 9 for the CMU database. It can be seen that the correlation filter

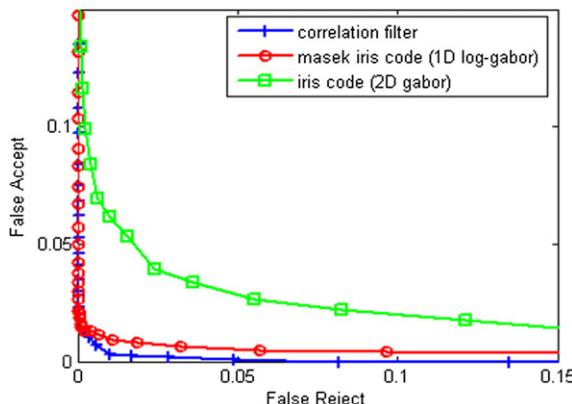


Fig. 9. ROC curves for the CMU iris image database.

method outperforms the Gabor wavelet encoding method in that correlation filters yield smaller FAR than the competing methods for the same FRR.

One point of interest in any ROC curve is the equal error rate (EER) where FAR = FRR. Correlation filters outperform the other methods, giving an EER of 0.61% for this database.

4. Bayesian graphical model for iris recognition

Unwrapped iris images from the same eye, acquired at two different time instants, can differ from each other significantly enough to lower their match score to cause the erroneous decision that they are not from the same eye. There are several possible reasons for the difference in appearance, for instance, when the pupil dilates or constricts the iris consequently compacts or stretches, respectively, changing its appearance. If this motion was perfectly linear along the radial direction, the iris segmentation process would normalize out this change. However, the motion is only approximately linear, and may not be limited to the radial direction, leading to minor deformations between two iris patterns from the same eye. In addition, even small segmentation errors can introduce nonlinear warping in the resulting mapped iris image. Figure 10 shows small regions in two images of the same iris in which minor pattern deformation is evident. Another cause for appearance difference is that the iris may be partially occluded by the eyelid in one image and not in the other image.

The iris matching algorithms we discussed in Sections 2 and 3 do not explicitly compensate for these nonlinear deformations and partial occlusions. To demonstrate the performance degradation due to these deformations and occlusions, we consider a sample pair of iris images (shown in Fig. 11) from the same eye. We define the Hamming similarity (HS) as (1-NHD) where NHD is the normalized Hamming distance between the corresponding binary codes. The HS for the Gabor codes in Fig. 11 is 0.68. Note that nonmatching iris pairs tend to have HS close to 0.5 (corresponding to random matching), and rarely exceeding 0.6. We introduced a range of artificial distortions to one of the patterns to observe their effect on the HS. The lower part of Fig. 11 displays the HS for this image pair as a function of slightly nonlinear deformation (vertical axis) and partial occlusion (horizontal axis). The deformation was implemented as a remapping along the radial direction:

$$I'(\rho, \theta) = I\left(\rho^{1+\alpha}, \theta\right), \quad (6)$$

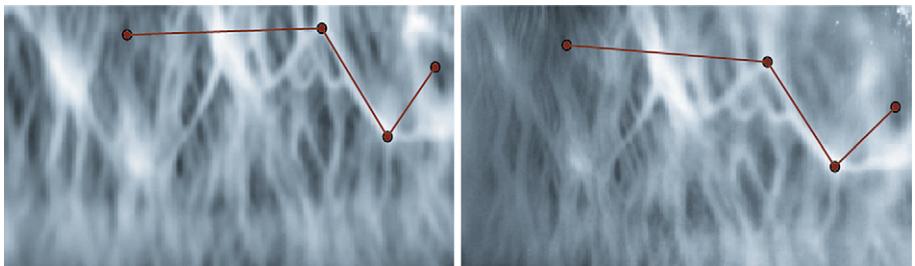


Fig. 10. Close-ups of segmented patterns from same eye (landmark points illustrate relative deformation).

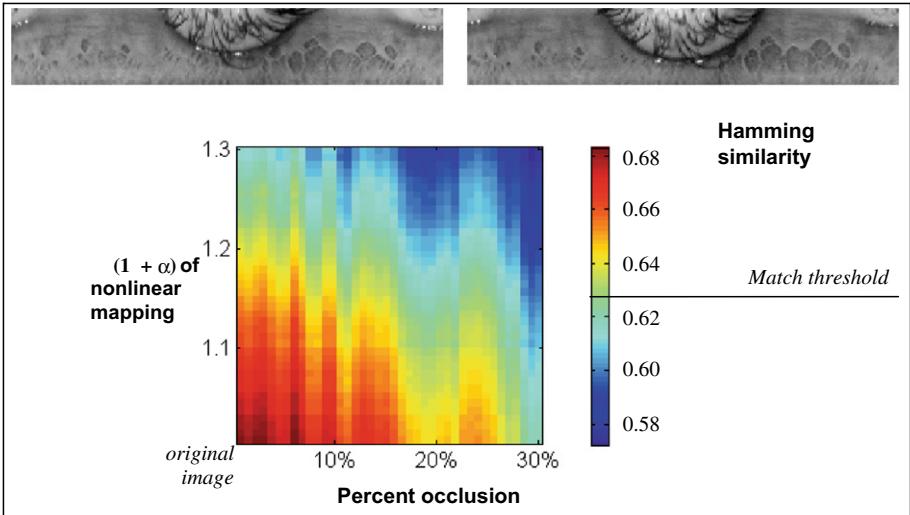


Fig. 11. Match score degradation under artificial deformation and occlusion.

where both the original segmented image $I(\rho, \theta)$ and the remapped image $I'(\rho, \theta)$ are defined for $\rho \in [0, 1]$. Partial occlusion was introduced by setting all values within an angular interval equal to a constant:

$$I'(\rho, \theta) = c \quad \text{for } \theta \in \left[\frac{\pi}{2} - \beta, \frac{\pi}{2} + \beta \right]. \quad (7)$$

The angular interval was centered around the top angle of the iris (where eyelid occlusion occurs most frequently) and β was set to give the desired percentage of occlusion. The match score threshold indicated in Fig. 11 was determined by taking the maximum of all experimental impostor HS scores. Figure 11 indicates that the conventional matching algorithm fails when α is bigger than 1.2 and occlusion is more than 20%.

Straightforward iris image matching without paying attention to possible nonlinear deformations and occlusions can lead to degraded similarity scores, particularly for authentic pairs. Thus false reject rates can be improved if we can estimate the presence and extent of such deformations between image pairs and use that information in estimating the match score. Fortunately, correlation filters can provide the clues for estimating these deformations and occlusions. By dividing the images into patches and cross-correlating corresponding patches, we can obtain information about the relative shifts between corresponding patches. Additionally, when an iris patch is occluded, the resulting correlation peak will be low. We next describe how these clues provided by patch cross-correlations can be used by a Bayesian graphical model to obtain a match score that takes the deformations and occlusions into account.

A high-level description of the problem of matching two iris images is shown in Fig. 12. Let R denote the stored reference image from class C and O denote the input probe iris image. Input O may or may not be a member of class C . Our goal is

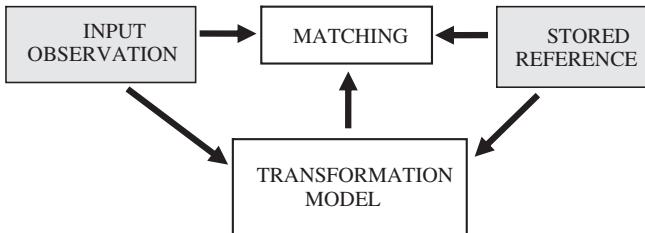


Fig. 12. Block diagram of image pair matching that takes deformations into account.

to obtain a similarity value $S(O, R)$ which quantifies the similarity between the two images. Ideally, $S(O, R)$ should be relatively large when $O \in C$ (referred to as an authentic case) and relatively small when $O \notin C$ (referred to as an impostor case). In the presence of occlusions and deformations (represented by the transformation T), we may be able to improve the matching process by first estimating T which describes the possibly nonlinear relationship found between registering images of the same class and then computing a modified similarity function $S(O, R; T)$ which takes into account the transformation. If T is estimated correctly for authentic comparisons, the modified similarity function should give a better indication of match.

The estimation of T suggests a probabilistic model for matching O and R . In this framework, the parameters in T are hidden variables which describe the coarse approximation of how iris images change from one authentic image to the next. Then the reference R , which defines the class (this could be the correlation filter template for a particular person's left iris, for example), along with hidden states T , can give a generative probability $p(O|R, T)$ of observing a new pattern O . Note that the latent variables in this model will describe both deformation and occlusion for improved iris matching.

4.1. MAP estimation of transformation

We start by assuming that observation O belongs to the class defined by reference R , and we estimate the most likely transformation parameter vector T . If the match assumption is true, this process should be correct for within-class transformations, increasing the similarity function value. If the match assumption is false (meaning, for example, there exists no relative deformation to register R and O), the transformation of an imposter image should not cause a significant increase in the similarity function value. This effect is demonstrated in our results, shown later.

Given the observed data O and reference template R , we try to find the best transformation parameters T to match the images

$$\hat{T} = \underset{T}{\operatorname{argmax}} p(T|O, R). \quad (8)$$

We treat the match assumption between O and R as implicit and do not state it explicitly in Eq. (8) or in any of the following notation. We can rewrite Eq. (8) as

$$\hat{T} = \underset{T}{\operatorname{argmax}} \frac{p(O|R, T)p(R, T)}{p(O, R)}. \quad (9)$$

Assuming that the transformation T and the reference template R are statistically independent and dropping unnecessary constants, this becomes

$$\hat{T} = \operatorname{argmax}_T p(O|R, T)p(T). \quad (10)$$

Here we are performing maximum a posteriori (MAP) estimation of T . The likelihood function on O is the generative probability, while the prior on T is the probability of the deformation itself. Intuitively, the MAP estimation process should favor smaller deformations (through higher prior probabilities) over more extreme deformations. That is, the probability of requiring small corrections (from distortion) to match authentic images is much higher than the probability of large adjustments.

The generative probability distribution $p(O|R, T)$ is defined over a very high-dimensional space (number of pixels or features being used in each image), which makes learning it very difficult. We simplify learning by setting the full distribution equal to a uni-variate distribution on the similarity metric:

$$p(O|R, T) = p(S(O, R; T)). \quad (11)$$

Here, the simplifying assumption is that the probability of generating the observed image O may be sufficiently determined by its similarity to the reference R (for example, if similarity is low, this will indicate a low probability of generating O from R using the transformation T). In our algorithm then, the similarity function serves a dual purpose: it allows us to model the generative probability during MAP estimation, and it is also used in calculating the final match score once the transformation has been estimated.

4.2. Estimation algorithm for iris images

For iris images, our transformation model T includes nonlinear in-plane deformations and partial occlusions. To implement the proposed matching algorithm, we must choose a way to parameterize the transformation. We use a model that consists of a set of local translations and occlusions. We divide the image plane into a set of N non-overlapping patches $\{Z_i\}$ with corresponding translation vectors $\{(\Delta x_i, \Delta y_i)\}$. The deformation is defined by the coordinate transform

$$(x, y) \rightarrow (x + \Delta x_i, y + \Delta y_i) \text{ for } (x, y) \in Z_i. \quad (12)$$

We define the vector field as a set of discrete random variables which take only integer values (i.e., we do not consider subpixel translation).

In addition, to model partial occlusion we define a corresponding set of binary occlusion variables $\{w_i\}$, one for each image patch. When $w_i = 1$, image region Z_i is occluded; when $w_i = 0$, it is not. Thus the transformation parameter vector T is as follows:

$$T = (\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, \dots, \Delta x_N, \Delta y_N, w_1, w_2, \dots, w_N)^t. \quad (13)$$

Because this transformation model partitions the image plane into a set of (assumed non-overlapping) regions, we can define the similarity function $S(O, R; T)$ in terms of local similarity functions. For each image patch, the local transformation is a single translation, along with a possible occlusion and patch-based cross-correlation

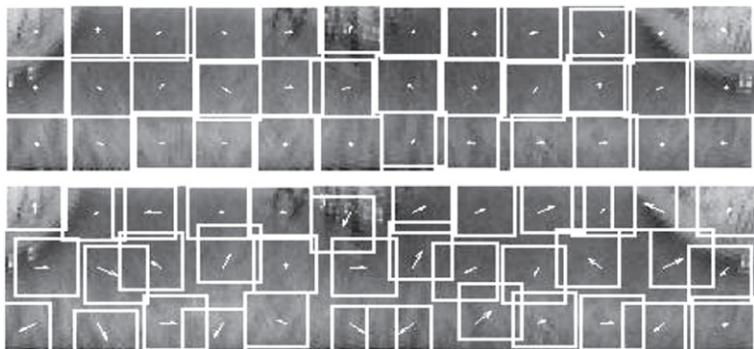


Fig. 13. Contrasting deformation fields. Top: an authentic match showing high likelihood. Bottom: a comparison with an impostor image showing lower likelihood.

outputs provide the information needed to estimate these transformation model parameters. Specifically, we design a correlation filter H_i for each image patch. If $w_i = 0$, the filter is designed using the i th sections of the reference images in R ; if $w_i = 1$, the filter is designed to respond strongly to a typical eyelid occlusion (the typical eyelid observation is learned directly from labeled data samples). Let $C_i(x, y)$ denote the correlation output when filter H_i is cross-correlated with image patch Z_i . So there are two possible realizations of each correlation output $C_i(x, y)$, determined by whether occlusion variable w_i equals one or zero.

We use the outputs of patch-based correlation filters to define the similarity function for all possible discrete values of the transformation vector T . Shown in Fig. 13 are two deformation fields that might align a query iris pattern to a template. The “boxes” are centered on the output of each patch correlation. The top image illustrates an alignment that is more likely to occur in authentic matches, while the alignment or “movements” of the patches to best fit the template to the query are much less reasonable. In order to learn the generative distribution $p(O|R, T) = p(S(O, R; T))$, we assume a uni-variate normal distribution on $S(O, R; T)$, and learn the mean and variance directly from sample data. Empirically, our iris data similarity scores seem to satisfy a normal distribution assumption. Likewise, we assume that the prior distribution $p(T)$ can be reasonably modeled as multi-variate normal distribution with mean at the zero vector (no transformation), and covariance matrix, Σ_T , learned from sample data using expectation maximization (EM). The learned weighting parameters α and β determine the importance of the magnitude of similarity per patch (correlation peak height) and the importance of the location of similarity (movement), respectively. These parameters are used to build the covariance:

$$\Sigma_T^{-1} = \begin{bmatrix} k_1(\alpha + \beta) & 0 & -\beta & 0 & 0 & 0 & 0 \\ 0 & k_1(\alpha + \beta) & 0 & -\beta & 0 & -\beta & 0 \\ -\beta & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & -\beta & 0 & k_j(\alpha + \beta) & 0 & -\beta & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & -\beta \\ 0 & -\beta & 0 & -\beta & 0 & k_N(\alpha + \beta) & 0 \\ 0 & 0 & 0 & 0 & -\beta & 0 & k_N(\alpha + \beta) \end{bmatrix}.$$

Note that the k_j term is the number of neighboring patches for the j th out of N patches. As expected, the covariance matrix enforces strong correlations between neighboring image patches, making it more likely that deformation and occlusion of nearby patches are consistent. The dependency of neighboring regions builds the undirected graphical model, known as a Markov Random Field, as a grid structure.

Due to the loops present within the formed grid, exact inference of the MAP estimate is intractable, however, approximate inference can be computed by using loopy belief propagation (Frey and Jojic, 2005). The details of this implementation can be found elsewhere in Thornton (2007) and Thornton et al. (2007). The final match score is the inner product of the result from loopy belief propagation with the outputs from the correlation filters (similarity function).

4.3. Numerical results

We first implemented the above matching algorithm for CMU iris data assuming deformations only. We measure performance by considering the improvement in recognition accuracy. We first perform the standard matching algorithm (without using the deformation model) and then compare it to the proposed algorithm. We used three images from each iris class as reference images. The equal error rate (EER) was 0.86% using conventional matching (i.e., without using the deformation model) whereas the EER improved to 0.29% using the Bayesian graphical model-based matching.

The EERs obtained for the CMU iris database are very good (smaller than 1%) even without using graphical models. To test the performance of the deformation model-based recognition, we tested this method on a more difficult image database called Face and Ocular Challenge Series (FOCS). Example FOCS images are shown in Fig. 14. As one can see, these images include periocular region including eyebrows,

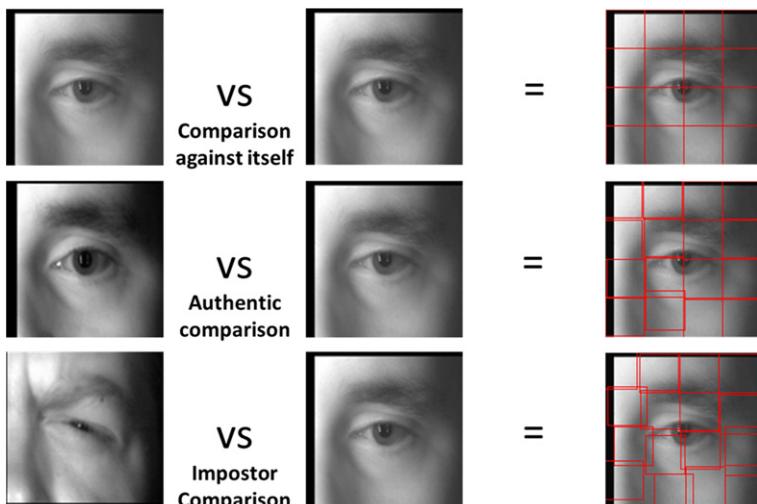


Fig. 14. An example of deformation estimation in matching.

skin near the eyes, and eye corners—not just the iris. It is anticipated that the inclusion of these additional cues can improve the matching performance.

The FOCS images were captured from moving subjects in an unconstrained environment showing drastic variations in sensor noise, illumination, gaze, and occlusion. In total, 9581 images (4792 left, 4789 right) over 136 subjects are separated by ocular region (left vs. left and right vs. right). A correlation filter is built for each image (with only the single image, no other authentic or impostor images are included and compared against each other image to generate two score matrices for the left (4792×4792) and right (4789×4789) ocular regions, respectively. The number samples per subject is not consistent, varying between 2 and 236 samples/subject, however 123 of the 136 subjects have at least 10 samples. Each image is down sampled from 750×600 pixels to 128×128 pixels for computation purposes, and divided into 16 non-overlapping patches (4×4 rectangular configuration). Testing was performed using 9 (3×3 configuration), 16 (4×4), 25 (5×5), 36 (6×6), 49 (7×7), and 64 patches (8×8) with the 4×4 configuration (of 16 total patches) showing the best results. Finally, each test used a randomly chosen set consisting of half of the left and right ocular images, respectively, to train the model.

An example application of this method can be seen in Fig. 14 (and in Fig. 13 when performing iris recognition) where comparing an image with itself registers perfectly (top), and comparing the same image with an authentic match yields small alignment corrections (shifts required by the red boxes to fit the authentic query to the original image). Moreover, comparing the same image with an impostor (third row of the figure), large shifts are required to best fit the two opposing users. As seen in Fig. 14, the benefit of using the model for recognition is the ability to account for deformations between two authentic images for better matching.

The ROC for the FOCS data (left and right ocular regions are shown separately) obtained using Bayesian graph model is shown in Fig. 15. The EERs obtained were 23.98% and 23.37% for the left and the right ocular regions, respectively.

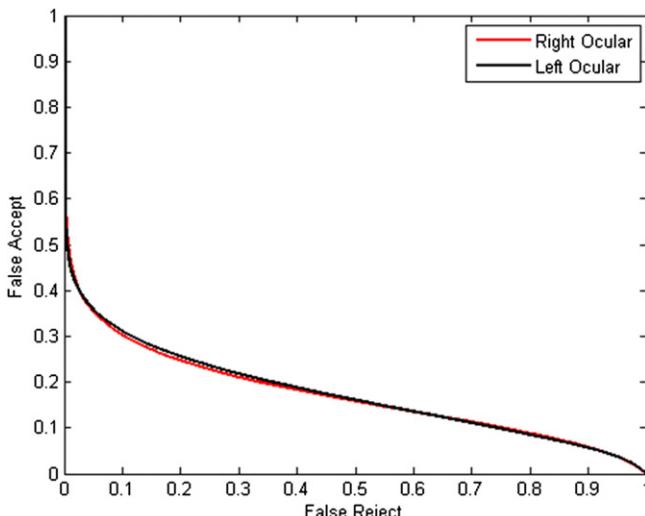


Fig. 15. ROC curve for FOCS data.

5. Summary

Iris recognition is of growing interest in the field of biometrics for human identification. We first summarized two techniques for iris recognition, namely Gabor wavelet-based iris encoding and the use of correlation filters. Although these methods work well for well-acquired iris images, the recognition rates degrade in more realistic acquisition conditions where the image appearance is affected by factors such as gaze angle, specular reflections, occlusions, and deformations. One way to match such challenging images is the use of Bayesian graphical models. These models can be built to allow for local deformations and occlusions and can provide a match score that takes these deformations and occlusions into account. Images are divided into non-overlapping patches and cross-correlations of patches can be used as inputs to the model. Cross-correlation peak values and peak locations are used by the model to learn parameters. Once the parameters for the model are estimated, the model can be used to produce a match score for a probe image and a template image. We showed matching results for a relatively easy iris image database and a more challenging ocular image database.

Acknowledgments

Part of this work was sponsored under IARPA BAA 09-02 through the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-10-2-0013. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of IARPA, the Army Research Laboratory, or the US Government.

References

- Adler, F.H., 1965. Physiology of the Eye. Mosby, St. Louis.
- Daugman, J.G., 1993. High confidence visual recognition of persons by a test of statistical independence. *IEEE Trans. Pattern Anal. Mach. Intell.* 15, 1148–1161.
- Dorairaj, V., Schmid, N., Fahmy, G., 2005. Performance evaluation of iris based recognition system implementing PCA and ICA encoding techniques. In: Proceedings of the SPIE Defense and Security Symposium.
- Frey, B.J., Jojic, N., 2005. A comparison of algorithms for inference and learning in probabilistic graphical models. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1392–1416.
- Huang, Y.P., Luo, S.W., Chen, E.Y., 2002. An efficient iris recognition system. In: Proceedings of the International Conference on Machine Learning and Cybernetics 1, pp. 450–454.
- Mahalanobis, A., Vijaya Kumar, B.V.K., Casasent, D., 1987. Minimum average correlation energy filters. *Appl. Opt.* 26, 3633–3640.
- Masek, L., Kovesi, P., 2003. MATLAB source code for a biometric identification system based on iris patterns. The University of Western Australia, The School of Computer Science and Software Engineering.
- Matey, J.R., Naroditsky, O., Hanna, K., Kolczynski, R., LoIacono, D.J., Mangru, S., Tinker, M., Zappia, T.M., Zhao, W.Y., 2006. Iris on the move: acquisition of images for iris recognition in less constrained environments. *Proc. IEEE* 94, 1936–1947.
- Refregier, Ph., 1990. Filter design for optical pattern recognition: multicriteria optimization approach. *Opt. Lett.* 15, 854–856.

- Thornton, J., 2007. Iris pattern matching: a probabilistic matching based on discriminative cues. Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, USA.
- Thornton, J., Savvides M., Vijaya Kumar, B.V.K., 2007. A unified Bayesian approach to deformed pattern matching of iris images. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 596–606.
- Vijaya Kumar, B.V.K., 1992. Tutorial survey of composite filter designs for optical correlators. *Appl. Opt.* 31, 4773–4801.
- Vijaya Kumar, B.V.K., Mahalanobis, A., Juday, R.D., 2005. Correlation Pattern Recognition. Cambridge University Press.
- Wildes, R.P., 1997. Iris recognition: an emerging biometric technology. *Proc. IEEE* 85, 1348–1363.
- Yu, L., Wang, K.Q., Wang, C.F., Zhang, D., 2002. Iris verification based on fractional Fourier transform. In: Proceedings of the International Conference on Machine Learning and Cybernetics, pp. 1470–1473.

Part IV: Document Analysis

This page is intentionally left blank

Learning Algorithms for Document Layout Analysis

Simone Marinai

Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze, Italy

Abstract

In this chapter we describe several approaches that have been proposed to use learning algorithm to analyze the layout of digitized documents. Layout analysis encompasses all the techniques that are used to infer the organization of the page layout of document images. From a physical point of view the layout can be described as composed by blocks, in most cases rectangular, that are arranged in the page and contain homogeneous content, such as text, vectorial graphics, or illustrations. From a logical point of view text blocks can have a different meaning on the basis of their content and their position in the page. For instance, in the case of technical papers blocks can correspond to the title, author, or abstract of the paper.

The learning algorithms adopted in this domain are often related to supervised classifiers that are used at various processing levels to label the objects in the document image according to physical or logical categories. The classification can be performed for individual pixels, for regions, or even for whole pages.

The different approaches adopted for using supervised classifiers in layout analysis are analyzed in this chapter.

Keywords: document image analysis and recognition, functional labeling, image segmentation, layout analysis, pixel classification

1. Introduction

Layout analysis is the process of identifying and recognizing the logical and physical structure of digitized documents. This is one of the main tasks in document image analysis and recognition whose aim is to extract the information from document images obtained, for instance, from books, journals, financial documents, and mail envelopes. The typical processing chain in most DIAR applications starts with pre-processing operations that are aimed at cleaning noisy images and are then followed

by layout analysis and by character/symbol recognition. Depending on the specific application domain, some post-processing operations can be considered as well (Marinai, 2008).

While the recognition of characters performed by Optical Character Recognition (OCR) engines is clearly suitable for the use of learning algorithms, the use of trainable classifiers to solve layout analysis problems is less intuitive.

1.1. Layout analysis approaches

The most important layout analysis techniques follow either a bottom-up or a top-down strategy even if several hybrid techniques have been adopted as well.

Bottom-up techniques identify the page structure with aggregation operations that start from the pixels that are often grouped in connected components (CCs). In the subsequent steps the connected components are first grouped in words that in turn are organized in text lines until homogeneous regions are found. Top-down techniques, on the opposite, start from the whole page that is recursively split into smaller sub-parts until the basic connected components are identified.

Examples of bottom-up algorithms are the Run-Length Smearing Algorithm (RLSA) (Wahl et al., 1982), the Docstrum algorithm by O'Gorman (1993), and algorithms based on Voronoi diagrams (Kise et al., 1998). One representative algorithm in the top-down category is the X-Y tree-based decomposition originally proposed by Nagy and Seth (1984) that has been subsequently extended by many authors, for instance, allowing cuts along horizontal and vertical ruling lines in addition to the standard cuts along white spaces (Cesarini et al., 1999).

Another important distinction that characterizes various approaches is related to the aim of the layout analysis performed. When the physical structure of the page should be inferred, these techniques are denoted as *physical layout analysis* or *document segmentation*. The objective of this analysis is therefore the identification of homogeneous regions containing either uniform text or illustrations, without attempting to assign any functional information to these regions. The latter problem is often referred to as *logical layout analysis* or *functional labeling*. Even in this case integrated techniques that combine the segmentation with the functional labeling have been proposed.

The actual text recognition is performed by Optical Character Recognition (OCR) engines that quite often rely on machine learning techniques to identify the symbol classes. While in most cases the text recognition follows the layout analysis step, it is also possible to revert the order of the two operations and identify the text regions on the basis of the features of the text recognized by OCR engines. The latter approach is particularly suitable for relatively clean documents, where the errors of the OCR tools are limited and for functional labeling tasks, where the recognition of the text content is essential to discover the logical role of text regions.

In this chapter we do not aim at providing a deep description of traditional techniques proposed to perform document structure analysis. Rather, we will emphasize the main strategies proposed to improve these techniques by using learning algorithms. Readers interested in standard techniques can refer for instance to Mao et al. (2003) where performance evaluation of document structure analysis algorithms is addressed or to other surveys (O'Gorman and Kasturi, 1995; Jain and Yu, 1998).

When designing segmentation algorithms, either bottom-up or top-down, some parameters should in most cases be considered in order to guide the algorithm. Suitable thresholds should be considered, for instance, to decide whether two connected components are expected to belong to the same word, or two words should be assigned to the same region. These parameters are frequently empirically hand-tuned during the algorithm design. In other cases the parameters are computed on the basis of some statistical analysis of representative documents performing a sort of manual training. Examples of the latter algorithms can be found in [Jain and Yu \(1998\)](#), [Haralick \(1994\)](#), [Lee et al. \(2001\)](#), and [Breuel \(2003\)](#).

1.2. Performance evaluation

Performance evaluation of page segmentation algorithms designed to work on a broad range of documents is difficult for two main reasons. First, some regions could be labeled with different classes in the context of different applications. For instance, one region containing some text printed over a background image could be labeled either as text or as image. Second, the regions of interest can have arbitrary shapes, but in most cases, to reduce the human effort, the labeling is made considering only areas delimited by upright rectangles. The latter problem is particularly critical for the training of pixel classification methods where pixels with a near-identical neighborhood could be labeled with different classes simply because the ground-truth information is not accurate.

In this chapter we will not discuss in detail performance evaluation of page segmentation algorithms. This task has been the subject of many works such as [Mao and Kanungo \(2001\)](#) and [Shafait et al. \(2007\)](#). Recent approaches for the performance evaluation of algorithms for region segmentation and classification are described in [Seo et al. \(2010\)](#).

1.3. Classification levels in layout analysis

Trainable classifiers can be used to perform several tasks in layout analysis, as we will discuss in the rest of this chapter. The main approaches are graphically summarized in Fig. 1 where different paths from the input document image (on the left) to output information (on the right) are shown and shortly summarized in the following.

Techniques related to pre-processing algorithms allow to assign a label to each foreground pixel with *pixel classification* algorithms. This classification is usually achieved by taking into account features, computed in a window surrounding the pixel of interest, that are expected to describe the image texture (Section 2). Areas in the page sharing the same class are then grouped together to provide the final segmentation.

To reduce the computational burden of the latter techniques one alternative is to compute the features from a window (usually larger than in pixel classification) that is regularly displaced over the page. This approach is sometimes referred to as *zone classification* and is discussed in Section 3.

The *classification of connected components*, that in most cases correspond to characters, can be used to locate text areas. This approach is particularly useful in some applications where natural scene images, obtained by mobile devices such

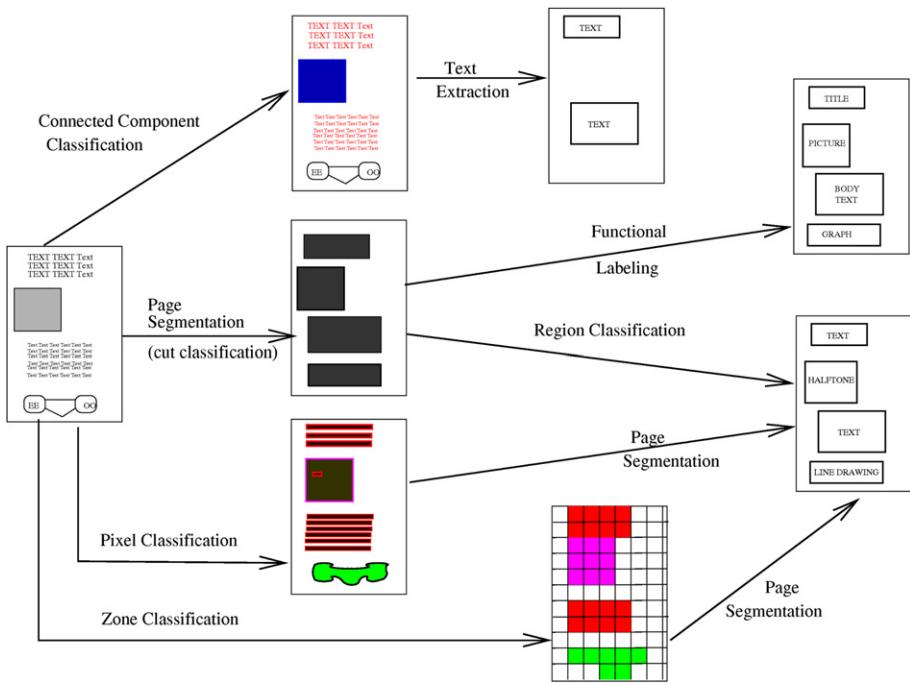


Fig. 1. Classifiers in layout analysis.

as smartphones and cameras, must be processed. These techniques are addressed in Section 4.

Some segmentation algorithms, such as the X-Y tree method, can use learning techniques to identify the most appropriate segmentation locations. In this case one trainable classifier is used to identify good segmentation points from a set of potential cuts (*cut classification*). These approaches are discussed in Section 5.

After identifying homogeneous regions it is possible to assign to each region one physical label that describes the type of objects contained. This process is made using *region classification* algorithms. The identification of text regions can be useful, for instance, to apply OCR tools only on these regions (Section 6).

When processing text regions with OCR tools it is also possible to figure out what is the purpose of the text region in the document with *functional labeling* operations. In so doing it is possible to identify specific document metadata such as the author and the title of scientific papers (Section 7).

In the rest of this chapter we discuss the classification tasks just summarized with a bottom-up organization. Section 2 is related to pixel classification and is then followed by zone classification (Section 3), connected component classification (Section 4), region classification (Section 6), text region segmentation (Section 5), and functional labeling (Section 7). Some conclusions are then drawn in Section 8.

2. Pixel classification

The lowest-level classification that can be performed when processing digital images is pixel classification. In this approach, a label is assigned to each pixel in the image. Image binarization is one of the simplest tasks that can be based on pixel classification. In this case the output class can be either foreground or background and the decision can be simply based on a comparison of the pixel's gray level with respect to a given threshold. The latter value can be hand-tuned during the system design and then fixed or can be learned by means of a training algorithm. While binarization is a traditional task in pre-processing, its extension to address additional classes brings to the use of pixel classification in layout analysis.

Figure 2 graphically summarizes the use of pixel classification. In the first, and most expensive, step a small window is moved over all the input pixels and the output of one trained classifier is assigned to the corresponding pixel in the output space. In Fig. 2 different labels are shown with different colors in the central image. For instance, red pixels correspond to locations where the classifier outputs a *text* label. As we can see from the example in some cases pixels with different labels are mixed together. To address the latter problem smoothing or grouping operations are often performed on the output of pixel classification to extract and label the layout regions. For instance, in Fig. 2 four regions are identified: two correspond to text areas while the remaining are labeled as halftone and line drawing, respectively.

Pixel classification for the binarization of document images is described in Islam et al. (2010) where an artificial neural network is used to classify pixels on the basis of the pixel intensity and of the mean and entropy of the pixels in a 3×3 window centered on the pixel to be labeled. It is interesting to notice that the performance of the proposed technique are evaluated considering the error rate of a commercial OCR on thresholded images obtained with various parameters.

Pixel classification for cleaning handwritten forms is described in Hidalgo et al. (2005) where a standard Multi-Layer Perceptron (MLP) trained with back-propagation is displaced across the input image considering as input a square region around the pixel to be labeled. The output unit is used in this case to estimate the desired gray level of the pixel to be cleaned.

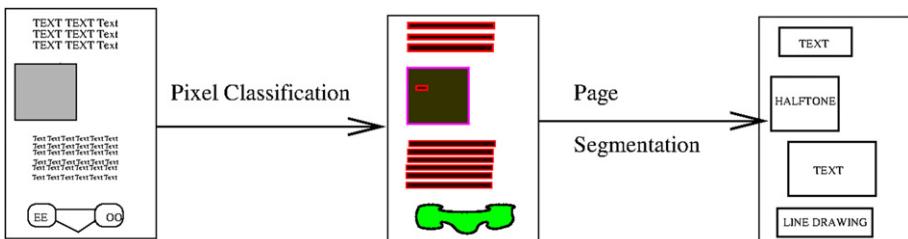


Fig. 2. Pixel classification.

In one related task the separation of printed text from overlapping handwriting is addressed in Peng et al. (2010) where pixels are aggregated in a bottom-up way by taking into account shape context features (Belongie et al., 2002). The pixel aggregation generates larger coarse regions whose pixels have coherent features. The document image foreground is subsequently represented with a graph where each node corresponds to a set of pixels; each edge of the graph connects nodes based on 4-neighbors lattice connectivity. During this aggregation, Markov Random Fields are used to model overlapped text and to separate *machine printed* text and *handwriting*. The latter problem is addressed by the authors also in Peng et al. (2010).

Due to the irregular shape of text captured by digital cameras, pixel classification is often adopted for the identification of text regions in natural scene images. For instance, in Clark and Mirmehdi (2000) suitable statistical measures are used to describe some properties of text areas that are invariant to orientation, scale, or color of the text. These features are then used as input of a three-layer MLP that is trained to classify the pixels as *text* or *non-text*.

In Ferreira et al. (2005) a method for texture characterization based on Gabor filters is proposed to address pixel classification applied to text detection from camera captured images. After computing the feature vectors for each pixel, one subset of pixels in the image is randomly selected and clustered with the k-means algorithm. The number of clusters, k , is empirically set to 3. The cluster whose centroid is closer to the origin of the feature space is then associated to the background, while the farthest cluster is used to label text pixels in the whole image.

Iterated pixel classification, that is broadly similar to cascade classifiers, is described in An and Baird (2008) for pixel classification. In this case the classes considered are *machine-print*, *handwriting*, *photograph*, and *background*. The method is based on a series of classifiers where subsequent classifiers are trained on the results of classification of the previous classifiers. The training data is decimated by randomly selecting only one out of every 9000 training samples. All the samples (pixels) are then re-classified. One important conclusion of this work is that the training in pixel classification is very sensitive to the approach adopted to build the ground-truth and in particular to the accuracy of ground-truth boundaries.

Chiu et al. (2010) propose the use of normalized graph cut (Shi and Malik, 2000) for clustering pixels belonging to different pictures in document images. Text regions are first identified by an OCR software and then removed from the document image. The remaining pixels, that are expected to belong to illustrations, are clustered so as to identify each illustration. The segmentation of illustrations is improved by looking for the image captions that are identified analyzing the text recognized by the OCR engine.

3. Zone classification

Pixel classification approaches suffer from two main problems. First, the high computational cost that is an obvious consequence of the classification of each pixel in the input image. Second, as discussed in An and Baird (2008), the training

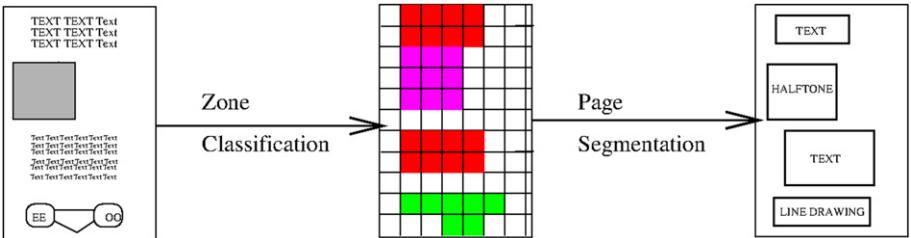


Fig. 3. Zone classification.

of pixel classification algorithms can be problematic in the frontier of regions where near equivalent inputs to the classifier can correspond to different classes. The use of pixel classification is suitable when handling irregularly-shaped zones and when the feature extraction and classifier execution are not too much demanding from the computational point of view. One possible solution to the previous problems, and in particular to the high computational cost, is achieved by using zone classification techniques. In this case one small window is moved across the image on pre-defined locations with low overlap between contiguous positions. In this case, with respect to pixel classification, the window is larger and the classification is not made for all the pixels. Moreover, the label provided by the classifier is assigned to all the pixels of the current window position and not only to the central pixel.

Figure 3 graphically summarizes the use of zone classification. In particular, the central image shows the, larger, blocks colored with the class provided by the classifier for each zone. The labeled regions are then obtained, in the right image, by merging neighboring zones having the same class.

An early approach to zone classification has been described in [Strouthopoulos and Papamarkos \(1998\)](#) where 8×8 zones in the page are classified as *text*, *graphics*, or *halftone* by means of artificial neural networks. To take into account zones partially overlapped to text areas, an additional class, *non-pure text*, is considered as well. The texture in each zone is described by a 34-dimensional feature vector whose size is reduced to 8 dimensions by means of PCA. The reduced feature vectors are then used to train an SOM with 8×8 topology that allows to discover some sub-classes that have, for instance, different font size and different writing style.

In [Micheal Baechler and Ingold \(2010\)](#) the segmentation of medieval manuscripts is addressed. An MLP is used to classify 32×32 zones into five classes: *text*, *comment*, *decoration*, *degradation*, and *background*. The features fed to the neural network are based on 10 mean colors computed by applying the k-means clustering on the pixels in the zone. Text areas are then merged together to obtain the desired line and block segmentation of textual regions.

In forensic document analysis [Chanda et al. \(2010\)](#) discuss a method for discriminating text and non-text zones in torn document fragments. The fragments include a limited amount of data and are arbitrarily oriented. In the proposed approach, $N \times N$ non-overlapping windows are moved across the image and then classified with a two-step approach. In the first step an SVM classifier is used to discriminate between *text* and *non-text* zones considering Gabor features. An SVM

classifier is used in the second step for segmenting *printed* and *handwritten* text. In this case chain-code-based features are used as input to the classifier.

In Gopalan and Manjula (2011) several texture-based features are computed from 8×8 blocks. The number of features is reduced with a feature selection algorithm proposed in the paper. The selected features are then used by an MLP classifier to discriminate between *text* and *non-text* regions in scene images.

The discrimination between *text* and *illustrations* is important to segment images of illuminated manuscripts (Grana et al., 2011). The basic feature used to extract the text is the autocorrelation matrix (the cross correlation of a signal with itself) that when applied to grayscale images provides a measure of the texture regularity. The autocorrelation is computed in square blocks in the input image. The block size is fixed on the basis of the scale at which the texture should be detected. This size is empirically fixed to a value that corresponds to the height of six text lines. The region classification is then obtained with an SVM with a radial basis kernel.

A different application of zone classification is described in Yang et al. (2004) where image enhancement is obtained by integrating zone classification with page segmentation. One page is first recursively decomposed by using quad-trees. Regions in the quad-tree are classified to identify the best image transformation to be used for enhancement (e.g., de-speckle, fill hole, and morphological operations). The input features are based on low-level characteristics, such as speckle and strokes and to connected component-based information.

Zone classification by means of an MLP has been used also to discriminate between *handwritten* and *printed* text in Koyama and Hirose (2008). In this paper, a fixed size zone (128×128) is moved across the page with an overlap of 8 pixels. In each position, features based on two-dimensional Fast Fourier Transform are used as input to an MLP-based classifier.

Text localization in scene images is described in Jung et al. (2009) where an initial text localization performed with heuristic methods is refined with zone classification. Each text candidate is first normalized to a height of 15 pixels. Subsequently, a 15×15 sliding window is horizontally displaced across the hypothesized text area and an SVM classifier is used to distinguish *text* and *non-text* areas. Labels provided for each window position are eventually combined to provide a unique classification for the whole region. The output score of the SVM is used to provide a measure of the closeness to text of an unknown area. Considering the SVM scores it is therefore possible to identify with more accuracy the left- and right-hand sides of the text region.

One key factor in the tuning of zone classification algorithms is the determination of a suitable size for the zones. When considering too large zones the results will be misleading since the zones contain in most cases mixed contents. On the other hand when using small zones a limited information is available for each position and the computational cost can be very high because the classification must be performed in several locations. To deal with these problems, in Won (2008) illustrations in documents are identified by integrating block-wise classification at various window sizes until single pixels are labeled. The initial optimal block size is found by analyzing the projection profiles and looking for blocks containing two text lines. The blocks are first classified on the basis of empirical decision rules based on the gray-level horizontal profiles. After this block classification, each boundary block (separating

groups of blocks with different labels) is further divided into four sub-blocks whose classes are computed again until the image boundaries are identified at the pixel level.

4. Connected component classification

In several scripts the connected components of clean documents coincide with characters. In other scripts the symbols are composed by multiple connected components. In the latter case by using smearing operations it is possible to join together sub-parts in a unique object. The connected components, or the larger objects, are then classified taking into account various features extracted from the object shape. Since connected components in text regions often correspond to individual characters, the majority of applications of connected component classification aim at identifying text from non-text objects.

Figure 4 graphically summarizes the use of connected component classification. In this case the central image shows colored connected components and the color corresponds to the classifier output. Red components are identified as text and by merging together neighboring components with the same color it is possible to identify text regions in the right part of the illustrations. By comparing Figs. 2–4 we can notice that a similar paradigm is applied in the three cases: some items are identified in the page and labeled with a trainable classifier. Clusters of objects sharing the same class are then merged together in order to define regions with homogeneous content. The main difference among the three approaches is the granularity of the labeling provided in the intermediate step that can correspond to pixels, to fixed-size zones, or to connected components.

In [Xi et al. \(2002\)](#) Chinese characters identified by merging connected components with smearing operations are classified in two categories: *body text* and *non-body text* components (including *headlines*, *graphics*, and *halftone pictures*) by using hard-coded rules that are not automatically learned with machine learning algorithms. On the basis of this classification, homogeneous text regions are extracted from the pages considering aggregation algorithms. In [Huang and Tan \(2007\)](#) a series of hand-tuned classifiers are applied to classify the connected components into *textual* components, *graphical* components, and *noise*. The classifiers take into account the

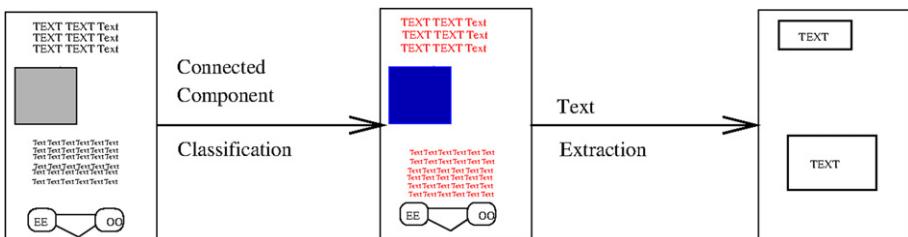


Fig. 4. Connected component classification.

size, the height and width, the height/width ratio, and the black pixel density of each connected component.

Other approaches use trainable classifiers and are therefore more relevant for this chapter. In [Bukhari et al. \(2010\)](#) an MLP is used to distinguish *text* and *non-text* connected components. The feature vector is in this case composed by shape and context information of each connected component. For shape features, each connected component is scaled to a 40×40 window. The other features are extracted from the surrounding context of the connected component.

A related approach is described in [Chen et al. \(2008\)](#) for identifying *handwritten* and *printed* text in scanned documents. The items to be classified, that can correspond to connected components or to larger layout elements, are identified with a bottom-up grouping of alpha shapes. A probabilistic classifier is then used to discriminate between four classes: *machine-printed*, *handwritten*, *noise*, and *unsure*. The latter class works like a reject class.

In [Haneda and Bouman \(2011\)](#) text segmentation from document images is used to perform document compression based on the Mixed Raster Content approach, where text glyphs are segmented and clustered to identify a set of prototypes. A preliminary segmentation is obtained by classifying partially overlapping blocks as *text* or *non-text*. In the second step connected components are classified to reduce false detections. The connected component classification is improved by means of a component-wise Markov Random Field (MRF) context model.

[Indermuhle et al. \(2010\)](#) compare techniques for *text* and *non-text* discrimination in handwritten documents. One of the techniques performs connected components classification by means of an SVM. The features considered are based on run-length histograms of black and white pixels along the four main directions. From the experimental comparison it turned out that the connected component classification outperformed the top-down approaches that are based on region classification (see Section 6).

Connected component labeling for text location in natural scene images is addressed in [Pan et al. \(2009\)](#) where Conditional Random Fields are used to provide a text or non-text label taking into account both component properties and binary neighboring component relationships. CRFs for pixel classification have been already discussed in Section 2.

Connected component classification is used also to identify the orientation of text lines in Roman and Indian documents (with vertical text lines). The approach proposed in [Rashid et al. \(2009\)](#) is based on the classification of connected components according to their expected orientation in the document. The page orientation is then determined via majority count. The classifier considers four classes corresponding to the main orientations (0° , 90° , 180° , and 270°) and is based on convolutional neural networks. The method is tested on one dataset containing Urdu documents.

In [Haneda and Bouman \(2009\)](#) connected component classification is applied to the analysis of camera-captured whiteboard notes. In this application, the notes are made by graph-based diagrams (mindmaps) where the overall graphical structure is first removed looking for the larger connected components. The remaining components are then classified by an MLP-based classifier into four classes: *text*, *lines*, *circles*, and *arrows*. At the end of the process, text components are grouped

together to form uniform text areas that can be processed by a suitable handwriting recognition engine. The features considered by the classifier are either standard statistical features (that are invariant in size and rotation) or intensities of gradient histograms.

One related problem is the classification of document patches that are obtained by merging together neighboring connected components with morphological operations. The patches correspond in most cases to words and are classified into *printed* text or *handwritten* annotations in Peng et al. (2010). In this paper an MLP-based classifier and one modified decision tree are considered. In the latter classifier all the training data are used for training at each node with different weights.

In Zheng et al. (2004), the connected components are first merged to form words considering spatial closeness. A Fisher classifier is then used to classify each word into *machine printed*, *text*, *handwriting*, or *noise*. To reduce the effects of misclassification between handwriting and noise the authors use contextual information in post-processing to refine the classification. This is achieved by incorporating the contextual information in the MRF models.

5. Text region segmentation

As previously discussed, document image segmentation algorithms are in most cases either based on split operations or on merge operations. Split operations are often found in top-down methods such as the X–Y tree algorithm where at each step one or more split positions are chosen among various alternatives. Merge operations are frequently encountered in bottom-up methods where contiguous objects (e.g., characters or words) can be merged together according to some criteria. In most cases, the choice of the operation to be performed is based on a heuristic tuning of the algorithm parameters (an implicit form of training, that is made by the system developer). In the approaches summarized in this section the decision is assessed on the basis of the output of one trainable classifier.

Figure 5 graphically summarizes the classification of cut points in page segmentation. In this case the input to the classifier are some features computed from zones in the input image that correspond to locations that could be used to split the image into homogeneous parts. In the left illustration dashed lines correspond to all the potential cuts evaluated by the classifier, while red lines (copied on the illustration on the right) identify cut locations confirmed by the classifier. As we can infer from this figure, the use of supervised classifiers for region segmentation is significantly different from the approaches described so far.

The identification of text blocks in a Bayesian framework is extensively addressed in Liang et al. (2000) and Liang et al. (2001). The text lines in the document are first extracted and represented with geometric features. The set of text lines is partitioned into sub-sets that correspond to text blocks that share formatting properties. One iterative technique is used to find a partitioning that maximizes the joint probability of splitting a sorted sequence of text lines into separate blocks.

A probabilistic approach to page segmentation is described in Shafait et al. (2008). The approach is designed to segment pages with Manhattan layouts where segmentation actions can be represented with an X–Y tree. A parametric model

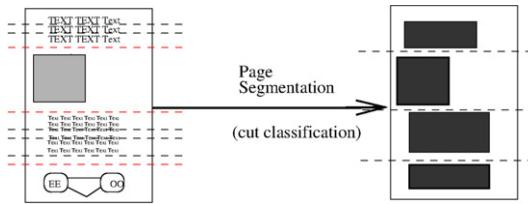


Fig. 5. Classification of segmentation points.

is trained to describe the variability in position and size of cuts on white spaces among various documents sharing the same layout. The distribution of parameters is estimated from the training set for each type of layout. The learned models are then matched on the page to be segmented and the best matching model is returned with the positions of best fit and the associated probability. The statistical layout matching is based on two main steps: first, a white space cover of page background is computed from the page; second, the white space rectangles are matched to model rectangles for different layout model components and the best fitting model is identified.

In Esposito et al. (2008) text blocks (in most cases corresponding to single words) are extracted from PDF documents and grouped together on the basis of a Multiple Instance Problem solved exploiting the kernel-based algorithm.

One related method is described in Gao et al. (2007) where each document image is first represented by a word-graph computed using a neighbor finding algorithm based on Voronoi tessellation. Edge weights measure how likely a pair of connected words belongs to the same zone. These weights are computed by an SVM classifier that identifies pairs of words to be merged. The features describe the difference in height between pairs of words and include 21 Harr-like filter responses from an image patch that is cut from the mid-point of an edge connecting two words. The patch size is twice the union of the two words bounding boxes. Two text boxes are then combined together if the patch comprised between the boxes is classified as text by the SVM.

A related approach is described in Wu et al. (2008) where the page segmentation is implemented through a series of split-or-merge operations. The segmentation is therefore considered as based on a series of binary decisions that are guided by SVM-based classifiers. The classifiers analyze the context of the expected segmentation position by computing features on the neighboring regions that are tested with respect to one possible split. The approach is tested on documents containing text with either horizontal or vertical reading order.

6. Region classification

In some segmentation algorithms the regions are identified before assigning to them a label. In this case a region classification step should be considered after region segmentation. In this section we discuss the approaches where the classification aims at identifying the general region content (e.g., text vs non-text) without considering

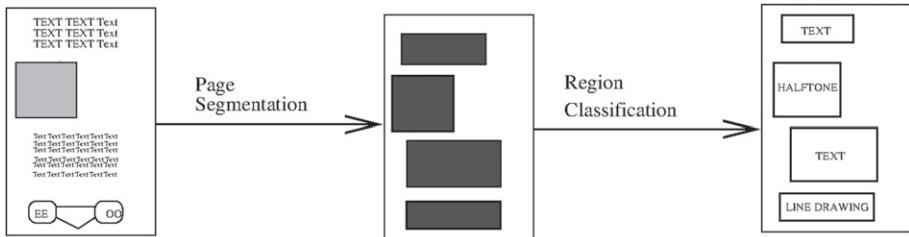


Fig. 6. Region classification.

the actual text in the region. On the opposite, when we aim at identifying the purpose of a text region on the basis of its contents (often recognized by an OCR engine), we deal with functional labeling applications that are discussed in the next section.

Figure 6 graphically summarizes the use of region classification. The input to the classifier are regions extracted with segmentation algorithms that could include also methods described in the previous sections. The classifier in this case works on features extracted from the whole region and that often includes statistical measures of the connected components or pixels in the region. Since the part of the document that is observed by the classifier is larger than in previous approaches it is possible to take advantage of the additional knowledge to provide a finer classification. Therefore, the output classes in region classification add, to the *text* vs *non-text* discrimination of the methods encountered so far, additional classes such as *halftone image*, *graphics*, *lines*. In other cases it is possible to distinguish between *English* and *Chinese* scripts.

Region classification was initially performed by using global features computed from the region as inputs to linear classifiers designed with user-defined parameters (Shih and Chen, 1996). In a similar way, when using trainable classifiers for region classification, some features are computed from each region and used as inputs to the classifiers.

One of the first approaches for zone classification is described in Chetverikov et al. (1996) where texture features are adopted together with a decision tree to classify the regions into *text* or *non-text*. The experiments are performed on the UW-III benchmark dataset.

Features for *non-text* region identification in newspapers are discussed in Andersen and Zhang (2003). First, a modified recursive X-Y cut algorithm is used to over-segment the image into regions. Since newspapers have a complex layout that includes overlapping and mixed text and non-text areas, the recursive X-Y cut algorithm can create homogeneous regions only by over-segmenting the image. An MLP is used to classify the over-segmented regions created by the X-Y cut algorithm. The features used are based on region information and on the connected components in the region. In total, almost 110 features have been tested. At the end of the process, the regions are coalesced into larger regions on the basis of the labels previously assigned and a second classification is performed to correct misclassifications.

Decision trees for region classification are described in Altamura et al. (2001) where rectangular regions are first identified by means of a variant of RLSA. The blocks returned by the segmentation algorithm are classified into: *text block*, *horizontal line*, *vertical line*, *halftone images*, and *graphics*. The features used to describe the blocks are based on the dimensions of the region, the number of black pixels, and the number of black-white transitions in the rows of the image regions. Other features are based on statistics obtained from the run-lengths in the region.

In Chi et al. (2003) regions in documents are first extracted with an enhancement of the background thinning approach proposed by Kise et al. (1996). These regions are then classified into text and picture with a multiple classifier system. In case of regions identified as text an artificial neural network is used to classify the script as *English* or *Chinese*. The features used by the neural classifier are adapted from those proposed by Imade and Tatsuta (1993). In particular, a small square window is moved across each region and in each position the histogram of gray levels and the histogram of the gradient vector directions are computed. These histograms are quantized into bins and the average values computed in each window in the region are used as features.

Duong et al. (2002) first extract regions of interest in grayscale document images. Extracted regions are classified as text or non-text areas by considering the entropy on horizontal projection histogram. The classifier used is based on Support Vector Machines.

A related application is described in Futrelle et al. (2003) where an SVM classifier is used to label diagrams extracted from PDF technical papers in two classes: “bar-graphs” and “other graphs.” The features considered as input to the classifier describe each diagram considering the number of rectangles, datapoints, lines or curves, categorical labels (non-numeric), short horizontal lines near a long vertical line, and short vertical lines near a long horizontal line. An extension of this method used to identify the type of illustrations in scientific PDF documents is described in Shao and Futrelle (2006). One MLP-based classifier is used to split the illustrations in five classes: *bar chart*, *tree/hierarchy*, *data graphs made by points, lines, or curves*. The features are based on the number of graphemes in each figure belonging to one of 16 grapheme classes (e.g., *line segment*, *curve*, *branch*, *data point*).

Indermuhle et al. (2010) compare techniques for text and non-text discrimination in handwritten documents. One of the two techniques is a top-down segmentation that adopts an SVM classifier to distinguish between text and non-text areas. The features are based on run-length histograms of black and white pixels along the horizontal, vertical, and the two diagonal directions. Another set of features is obtained by considering the width and height distributions of the connected components in the zone. The last group of features is based on a two-dimensional histogram of the joint distribution of widths and heights and a histogram of the nearest neighbor distances of the connected components. In total 152 features are used as input to the SVM classifier.

The classification of zones into printed and handwritten Arabic text is described in Kumar et al. (2011) where regions are first identified by means of the Voronoi ++ method (Agrawal and Doermann, 2009). For each region, local contour features, called k-adjacent segments, are computed considering the triplets of neighboring segments obtained by fitting segments on edges of connected components. Each

zone is described by means of the occurrences of codebook triplets in the zone where the codebooks are computed (for printed and handwritten regions) with k-means. This description is then considered as input to an SVM classifier to discriminate between printed and handwritten zones.

Zone content classification by using decision trees is described by Wang et al. (2006). A 25-dimensional feature vector is first extracted from each zone and then a trained decision tree is used to distinguish among nine classes. The features extracted from each zone are based on run-lengths computed among four main directions in the zone, spatial features, autocorrelation features, background features, and other information with a total of 69 measures. The classes, corresponding to the labels in the UW-III benchmark dataset are: *large text*, *small text*, *math*, *table*, *halftone*, *map/drawing*, *ruling*, *logo*, and *others*.

7. Functional labeling

Functional labeling aims at assigning a logical class to regions. In most cases this task is performed on regions previously identified as text and whose content has been recognized by an OCR engine. These techniques are very close to information extraction methods. It is also possible to deal with digital-born documents where the text content is already explicitly encoded in the document.

Figure 7 graphically summarizes the most common way to perform functional labeling. In analogy with region classification, regions extracted from the input image are labeled with a class that describes its meaning or function in the document. In this example, the top text region is labeled as *title* while the lower text region is labeled as *body text*. It is also possible to use pixel or connected component classification to perform functional labeling as we summarize in the following.

Conditional Random Fields (CRFs) have been used to perform functional labeling by mean of pixel classification. In Montruil et al. (2009) the physical and logical layouts in handwritten letters are extracted by CRFs that are used to label the pixels according to layout elements. By using CRFs it is possible to take into account the relationship of one class with respect to those of its neighboring elements. Examples of classes considered are *background*, *sender*, *date*, *address*, *object*, *open*, *body text*, and *signature*. Conditional Random Fields for information extraction

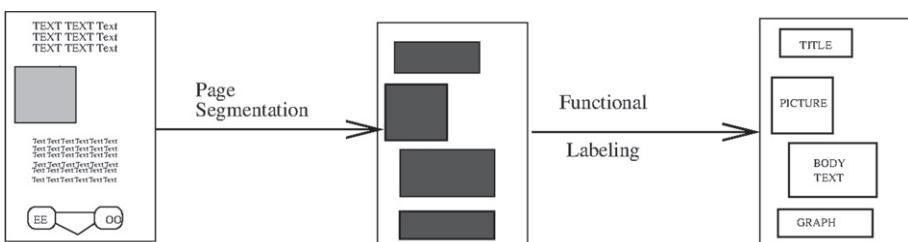


Fig. 7. Functional labeling of text regions.

are analyzed also in [Paass and Konya \(2009\)](#) together with other applications of machine learning for document structure recognition.

One of the first approaches that used connected component classification in layout analysis is described in [Iwane et al. \(1993\)](#) where logical labels are assigned to connected components considering nine features: the coordinate and size of the connected component, the line height, line spacings (above and below), and margins (left and right). The latter values are computed considering the estimated text line that is found by inspecting the horizontal projection profile overlapping with the connected component. A K-NN classifier is adopted considering the Euclidean distance of the feature vector with a collection of labeled patterns. The 14 classes considered cover most of the information appearing in title pages such as *title*, *authors*, *abstract*, *page number*.

One accurate comparison of methods for logical structure analysis has been presented in [Lee et al. \(2003\)](#). According to this analysis, most approaches can be classified as belonging to model-matching methods (based in some cases on learning) and to syntactic methods (that frequently use document models or grammars).

Logical layout analysis is performed in [Bayer \(2003\)](#) considering a document model and inference rules. *Sender* and *receiver* of business letters are considered in this case.

[Belaid and Rangoni \(2008\)](#) describe Perceptive Structured Neural Networks to perform logical structure analysis. In this architecture, each neuron corresponds to an interpretable concept and is attached to an element of the logical structure. In particular, the first layer is composed by physical features, whereas the following layers introduce concepts that are finer in the first layers and more general in the latest.

8. Conclusion

In this chapter we discussed and compared several strategies where learning algorithms are integrated with image processing techniques to perform layout analysis of digitized documents. Trainable classifiers can be used at several levels in layout analysis ranging from the labeling of pixels to the labeling of regions in the page. For each category of techniques, the main approaches are summarized emphasizing the features that are used as input to the classifiers and the classes that are considered as output of the classifier. Due to the large number of applications addressed by the various methods it is not possible to make a systematic and meaningful comparison of the results achieved by the alternative approaches.

References

- Agrawal, M., Doermann, D.S., 2009. Voronoi++: a dynamic page segmentation approach based on Voronoi and docstrum features. In: Proceeding of ICDAR, pp. 1011–1015.
- Altamura, O., Esposito, F., Malerba, D., 2001. Transforming paper documents into XML format with WISDOM++. Int. J. Doc. Anal. Rec. 4 (1), 2–17.
- An, C., Baird, H.S., 2008. The convergence of iterated classification. In: International Workshop on Document Analysis Systems, pp. 663–670.

- Andersen, T., Zhang, W., 2003. Features for neural net based region identification of newspaper documents. In: Proceeding of ICDAR, pp. 403–407.
- Bayer, T.A., 2003. Understanding structured text documents by a model based document analysis system. In: Proceeding of ICDAR, pp. 448–453.
- Belaid, A., Rangoni, Y., 2008. Structure extraction in printed documents using neural approaches. In: Marinai, S., Fujisawa, H. (Eds.), Machine Learning for Document Analysis and Recognition, pp. 21–43.
- Belongie, S., Malik, J., Puzicha, J., 2002. Shape matching and object recognition using shape contexts. IEEE TPAMI 24 (4), 509–522.
- Breuel, T.M., 2003. An algorithm for finding maximal whitespace rectangles at arbitrary orientations for document layout analysis. In: Proceeding of ICDAR, pp. 66–70.
- Bukhari, S.S., Azawi, M.I.A.A., Shafait, F., Breuel, T.M., 2010. Document image segmentation using discriminative learning over connected components. In: International Workshop on Document Analysis Systems, pp. 183–190.
- Cesarini, F., Gori, M., Marinai, S., Soda, G., 1999. Structured document segmentation and representation by the modified X-Y tree. In: ICDAR, pp. 563–566.
- Chanda, S., Franke, K., Pal, U., 2010. Document-zone classification in torn documents. In: International Conference Frontiers in Handwriting Recognition, pp. 25–30.
- Chen, J., Saund, E., Wang, Y., 2008. Image objects and multi-scale features for annotation detection. In: International Conference on Pattern Recognition, pp. 1–5.
- Chetverikov, D., Liang, J., Komuves, J., Haralick, R.M., 1996. Zone classification using texture features. In: International Conference on Pattern Recognition, pp. 676–680.
- Chi, Z., Wang, Q., Siu, W.-C., 2003. Hierarchical content classification and script determination for automatic document image processing. Pattern Recognition 36 (11), 2483–2500. <<http://www.sciencedirect.com/science/article/pii/S0031320303001286>>.
- Chiu, P., Chen, F., Denoue, L., 2010. Picture detection in document page images. In: ACM Symp. Document Engineering, pp. 211–214.
- Clark, P., Mirmehdi, M., 2000. Finding text regions using localised measures. In: BMVC, pp. 675–684.
- Duong, J., Coté, M., Emptoz, H., 2002. Feature approach for printed document image analysis, 159–167. <<http://dl.acm.org/citation.cfm?id=671246>>.
- Esposito, F., Ferilli, S., Basile, T.M., Mauro, N.D., 2008. Machine learning for digital document processing: from layout analysis to metadata extraction. In: Marinai, S., Fujisawa H. (Eds.), Machine Learning for Document Analysis and Recognition, pp. 105–138.
- Ferreira, S., Garin, V., Gosselin, B., 2005. A text detection technique applied in the framework of a mobile camera-based application. In: CBDAR, pp. 133–139.
- Futrelle, R.P., Shao, M., Cieslik, C., Grimes, A.E., 2003. Extraction, layout analysis and classification of diagrams in PDF documents. In: Proceeding of ICDAR, pp. 1007–1013.
- Gao, D., Wang, Y., Hindi, H., Do, M., 2007. Decompose document image using integer linear programming. In: Proceeding of ICDAR, pp. 397–401.
- Gopalan, C., Manjula, D., 2011. Statistical modeling for the detection, localization and extraction of text from heterogeneous textual images using combined feature scheme. Signal Image Video Process. 5, 165–183.
- Grana, C., Borghesani, D., Cucchiara, R., 2011. Automatic segmentation of digitalized historical manuscripts. Multimedia Tools Appl. 55 (3), 483–506.
- Haneda, E., Bouman, C.A., 2009. Text segmentation for MRC document compression. J. Universal Comput. Sci. 15, 3307–3324.
- Haneda, E., Bouman, C.A., 2011. Text segmentation for MRC document compression. IEEE Trans. Image Process. 20 (6), 1611–1626.
- Haralick, R.M., 1994. Document image understanding: geometric and logical layout. In: Proceedings of IEEE Conference Computer Vision and Pattern Recognition, pp. 385–390.
- Hidalgo, J.L., Espana, S., Castro, M.J., Pérez, J.A., 2005. Enhancement and cleaning of handwritten data by using neural networks. In: IBPRIA, pp. 376–383.
- Huang, W., Tan, C.L., 2007. A system for understanding imaged infographics and its applications. In: ACM Symp. Document Engineering, pp. 9–18.
- Imade, S., Tatsuta, S., Wada, T., 1993. Segmentation and classification for mixed text/image documents using neural networks. In: ICDAR, vol. 93, pp. 930–934.

- Indermuhle, E., Bunke, H., Shafait, F., Breuel, T., 2010. Text versus non-text distinction in online handwritten documents. In: SAC, pp. 3–7.
- Islam, M.J., Ahmadi, M., Sid-Ahmed, M.A., Alginahi, Y.M., 2010. Optimal parameter selection technique for a neural network based local thresholding method. *J. Pattern Recog. Res.* 1, 69–94.
- Iwane, K., Yamaoka, M., Iwaki, O., 1993. A functional classification approach to layout analysis in document images. In: Proceeding of ICDAR, pp. 778–781.
- Jain, A.K., Yu, B., 1998. Document representation and its application to page decomposition. *IEEE TPAMI* 20 (3), 294–308.
- Jung, C., Liu, Q., Kim, J., 2009. Accurate text localization in images based on SVM output scores. 27, 1295–1301. <http://dx.doi.org/10.1016/j.imavis.2008.11.012>. <<http://www.sciencedirect.com/science/article/pii/S0262885608002515>>.
- Kise, K., Yanagida, O., Takamatsu, S., 1996. Page segmentation based on thinning of background. In: International Conference on Pattern Recognition, pp. 788–792.
- Kise, K., Sato, A., Iwata, M., 1998. Segmentation of page images using the area Voronoi diagram. *Comput. Vis. Image Understand.* 70, 370–382.
- Koyama, J., Hirose, A., Kato, M., 2008. Local-spectrum-based distinction between handwritten and machine-printed characters, 1021–1024. <http://dx.doi.org/10.1109/ICIP.2008.4711931>.
- Kumar, J., Prasad, R., Cao, H., Abd-Almageed, W., Doermann, D., Natarajana, P., 2011. Shape codebook based handwritten and machine printed text zone extraction. In: DRR, pp. 1–10.
- Lee, S.-W., Member, S., Ryu, D.-S., 2001. Parameter-free geometric document layout analysis. *IEEE TPAMI* 23 (11), 1240–1256.
- Lee, K.-H., Choy, Y.-C., Cho, S.-B., 2003. Logical structure analysis and generation for structured documents: a syntactic approach. *IEEE Trans. Knowl. Data Eng.* 15 (5), 1277–1294.
- Liang, J., Phillips, I.T., Haralick, R.M., 2000. Consistent partition and labelling of text blocks. *PAA* (3), 196–208.
- Liang, J., Phillips, I.T., Haralick, R.M., 2001. An optimization methodology for document structure extraction on latin character documents. *IEEE TPAMI* 23 (7), 719–734. <http://dx.doi.org/10.1109/34.935846>.
- Mao, S., Kanungo, T., 2001. Empirical performance evaluation methodology and its application to page segmentation algorithms. *IEEE TPAMI* 23 (3), 242–256.
- Mao, S., Rosenfeld, A., Kanungo, T., 2003. Document structure analysis algorithms: a literature survey. In: DRR, pp. 197–207.
- Marinai, S., 2008. Introduction to document analysis and recognition. In: Marinai, S., Fujisawa, H. (Eds.), *Machine Learning for Document Analysis and Recognition*, pp. 1–20.
- Micheal Baechler, J.-L.B., Ingold, R., 2010. Semi-automatic annotation tool for medieval manuscripts. In: International Conference Frontiers in Handwriting Recognition, pp. 182–187. <http://dx.doi.org/10.1109/ICFHR.2010.36>.
- Montruil, F., Grosicki, E., Heutte, L., Nicolas, S., 2009. Unconstrained handwritten document layout extraction using 2D conditional random fields. In: Proceeding of ICDAR, pp. 853–857.
- Nagy, G., Seth, S., 1984. Hierarchical representation of optically scanned documents. In: ICPR, vol. 84, pp. 347–349.
- O’Gorman, L., 1993. The document spectrum for page layout analysis. *IEEE TPAMI* 15 (11), 1162–1173.
- O’Gorman, L., Kasturi, R., 1995. *Document Image Analysis*. IEEE Computer Society Press, Los Alamitos, California.
- Paass, G., Konya, I., 2009. Machine learning for document structure recognition. In: Mehler, A., K++hnberger, K.-U., Lobin, H., L++ngen, H., Storrer, A., Witt A. (Eds.), *Modeling, Learning, and Processing of Text Technological Data Structures*. [<http://dx.doi.org/10.1007/978-3-642-22613-7_12>](http://dx.doi.org/10.1007/978-3-642-22613-7_12). <http://link.springer.com/chapter/10.1007%2F978-3-642-22613-7_12?LI=true#>.
- Pan, Y.-F., Hou, X., Liu, C.-L., 2009. Text localization in natural scene images based on conditional random field. In: Proceeding of ICDAR, pp. 6–10.
- Peng, X., Setlur, S., Govindaraju, V., Sitaram, R., 2010. Overlapped text segmentation using markov random field and aggregation. In: International Workshop on Document Analysis Systems, pp. 129–134.
- Peng, X., Setlur, S., Govindaraju, V., Sitaram, R., 2010. Text separation from mixed documents using a tree-structured classifier. In: International Conference on Pattern Recognition, pp. 241–244.

- Rashid, S.F., Bukhari, S.S., Shafait, F., Breuel, T.M., 2009. A discriminative learning approach for orientation detection of Urdu document images. In: International Multitopic Conference, pp. 1–5.
- Seo, W., Agrawa, M., Doermann, D., 2010. Performance evaluation tools for zone segmentation and classification (PETS). In: International Conference on Pattern Recognition, pp. 503–506.
- Shafait, F., Keysers, D., Breuel, T.M., 2007. Performance evaluation and benchmarking of six page segmentation algorithms. *IEEE TPAMI* 30 (6), 941–954.
- Shafait, F., van Beusekom, J., Keysers, D., Breuel, T.M., 2008. Structural mixtures for statistical layout analysis. In: International Workshop on Document Analysis Systems, pp. 415–422.
- Shao, M., Futrelle, R.P., 2006. Recognition and classification of figures in PDF documents, pp. 231–242. http://dx.doi.org/10.1007/11767978_21. <<http://dl.acm.org/citation.cfm?id=2102482>>.
- Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. *IEEE TPAMI* 22 (8), 888–905.
- Shih, F.Y., Chen, S.S., 1996. Adaptive document block segmentation and classification. *IEEE Trans. SMC* 26 (5), 797–802.
- Strouthopoulos, C., Papamarkos, N., 1998. Text identification for document image analysis using a neural network. *Image Vis. Comput.* 16 (12/13), 879–896.
- Wahl, F., Wong, K., Casey, R., 1982. Block segmentation and text extraction in mixed text/image documents. *Comput. Graphics Image Process.* 20, 375–390.
- Wang, Y., Phillips, I.T., Haralick, R.M., 2006. Int. J. Doc. Anal. Rec. 39, 57–73.
- Won, C.S., 2008. Image extraction in digital documents. *J. Electron. Imag.* 17 (3).
- Wu, C.-C., Chou, C.-H., Chang, F., 2008. A machine-learning approach for analyzing document layout structures with two reading orders. *Pattern Recognition* 41, 3200–3213.
- Xi, J., Hu, J., Wu, L., 2002. Page segmentation of chinese newspapers. *Pattern Recognition* 35, 2695–2704.
- Yang, Y., Summers, K., Turner, M., 2004. A text image enhancement system based on segmentation and classification methods, 33–40. <http://dx.doi.org/10.1145/1031442.1031448>. <<http://dl.acm.org/citation.cfm?id=1031448>>.
- Zheng, Y., Li, H., Doermann, D., 2004. Machine printed text and handwriting identification in noisy document images. *IEEE TPAMI* 26 (3), 337–353. <http://dx.doi.org/10.1109/TPAMI.2004.1262324>.

This page is intentionally left blank

Hidden Markov Models for Off-Line Cursive Handwriting Recognition

Andreas Fischer, Volkmar Frinken, and Horst Bunke*

*University of Bern, Institute of Computer Science and Applied Mathematics,
Neubrückstrasse 10, 3012 Bern, Switzerland*

Abstract

Hidden Markov models nowadays belong to the most widely used statistical models for the challenging task of handwriting recognition in document images. In this chapter, we describe the predominant application given by segmentation-free recognition of handwritten text lines in presence of large vocabularies and multiple writers. A review of the state of the art is provided for feature sequence extraction, character appearance modeling, and language modeling. In particular, we comment on the typical parametrization of hidden Markov models for appearance modeling and discuss efficient dynamic programming solutions for training and recognition. Besides the review of appearance and language modeling, this chapter also provides an introduction to confidence modeling, which is an important topic of current research. Confidence models estimate the reliability of a recognition result that can be used for diverse applications including rejection of unreliable results, writer identification and verification, combination of multiple classifiers, and keyword spotting. Finally, some trends and challenges for future research are identified.

Keywords: statistical pattern recognition, hidden Markov models, handwriting recognition, language modeling, confidence modeling

1. Introduction

Since more than half a century, automatic handwriting recognition has been an intriguing and still largely unsolved problem in the discipline of pattern recognition (Lorette, 1999). The goal to match or even surpass the impressive ability of humans

*Corresponding Author

to recognize handwritten text, produced by different writers with greatly varying writing styles, is far from being reached, although significant progress has been achieved (Bunke and Varga, 2007). Besides the scientific interest, there are also many commercial applications that have been a driving force behind the development of new recognition systems. Examples include the automatic reading of postal addresses (Srihari, 2000), bank cheques (Impedovo et al., 1997), forms (Srihari et al., 1996), letters sent to companies (Rodriguez and Perronnin, 2009),¹ and, more recently, historical documents for integration into digital libraries (Antonacopoulos and Downton, 2007).

Commercial handwriting recognition systems that achieve near-perfect recognition performance are only available for restricted tasks with a small vocabulary, e.g., for bank check reading (Gorski et al., 2001). Also, there are commercial products available for *on-line* handwriting recognition (Plamondon and Srihari, 2000), where temporal information is available about the writing process based on special input devices, e.g., for Personal Digital Assistants (PDAs). In contrast, *off-line* recognition is based solely on handwriting images. Because of the lack of temporal information, it is considered to be a harder task than on-line recognition. For multiple writers and large vocabularies underlying natural language, the accuracy of an automatic transcription of off-line handwritten text is far from being perfect. An intriguing problem for cursively written, connected handwriting arises from the fact that text segmentation into characters is not feasible without recognition and character recognition is not feasible without segmentation. Also known as Sayre's paradox (Sayre, 1973), this contradiction renders standard Optical Character Recognition (OCR) systems, which are based on an accurate character segmentation, infeasible for most handwriting recognition tasks.

The application of hidden Markov models (HMMs) for off-line handwriting recognition has proven to be one of the few approaches that can cope with the difficult recognition conditions mentioned. Originally proposed for the task of speech recognition (Rabiner, 1989), these powerful statistical models could also be adopted for handwriting recognition. In fact, HMMs are probably the most widely used statistical models for handwriting recognition nowadays, and in the past two decades they have played an important role for the improvements achieved in the field (Ploetz and Fink, 2009). One of the most important features of HMM-based recognition is that segmentation and recognition can be achieved at the same time, i.e., no segmentation of the text images into characters is needed prior to recognition. Also, the availability of efficient algorithms for learning and recognition is of great importance for the success of HMM-based handwriting recognition.

In this chapter, we review the state of the art of HMM-based handwriting recognition and show trends and challenges for future research. Hereby, we focus on the predominant application given by segmentation-free recognition of handwritten text lines for large vocabularies and multiple writers. Based on character models, the presented methods are applicable to any alphabetical language. Not included in the

¹Despite the increasing amount of Email correspondence, companies still receive large amounts of handwritten letters everyday.

review are segmentation-based approaches as well as single character recognition. Single word recognition is treated as a special case of text line recognition. The description of the methods is guided by a statistical point of view. For a survey on current systems and datasets, we refer to [Ploetz and Fink \(2009\)](#).

The chapter is structured as follows. First, the serialization of text line images and feature extraction for HMM-based recognition is addressed in Section 2. Then, HMM-based text line recognition is discussed step by step in Section 3. In this section, the typical topology and parameterization of character HMMs is introduced in Section 3.1.1. Next, algorithms for appearance-based recognition are discussed in Section 3.1.2. The integration of language models is presented afterward in Section 3.2. In addition to these well-established methods, an introduction to confidence modeling, an emerging topic of research, is provided in Section 3.3. Finally, conclusions are drawn in Section 4 together with a discussion of open problems.

2. Serialization of handwriting images

HMMs are a powerful statistical tool for modeling a sequence of events. Hence, for HMM-based recognition, handwriting images have to be serialized in a first step by extracting a linear signal from the two-dimensional data. From a signal theory oriented point of view, it would be desirable to reconstruct the original movements of the writing instrument from a given handwriting image. However, in contrast to on-line handwriting recognition, where temporal information about the writing process is available due to special input devices ([Plamondon and Srihari, 2000](#)), the reconstruction of this on-line information from a scanned or photographed image is a very challenging problem. Although several attempts have been made in this direction ([Boccignone et al., 1993; Bunke et al., 1997; Doermann and Rosenfeld, 1992; Plamondon and Privitera, 1999](#)), mainly in the 1990s, a workaround based on a sliding window approach, which works very well in practice, has emerged as the predominant method in the last decade ([Bertolami and Bunke, 2008; Brakensiek and Rigoll, 2004; El-Hajj et al., 2005; Espana-Boquera et al., 2011; Schambach, 2008; Vinciarelli et al., 2004; Wienecke et al., 2005](#)).

In the sliding window approach, a small analysis window is moved in writing direction over the image of an individual text line and a set of features, typically real valued, are extracted at each window position. This results in a feature vector sequence description $\mathbf{x} = x_1, \dots, x_T$ of the text line image with $x_i \in \mathbb{R}^n$ for T window positions. This vector sequence can then be used for HMM-based recognition (see Section 3).

Before serialization, an input document image has to be segmented into text lines. On the one hand, this challenging task typically involves layout analysis to identify text blocks consisting of consecutive lines, and on the other hand, segmentation methods are needed for separating the text lines from each other within the text blocks. For a survey on text line extraction methods, we refer to [Likhman-Sulem et al. \(2007\)](#). Results of a recent competition were reported by [Gatos et al. \(2010\)](#).

Typically, the text line images obtained by text line extraction contain different variations in the appearance of the handwriting that impose difficulties for feature

extraction. First, depending on the input device, background noise may occlude the handwriting foreground. Also, the general orientation and size of the handwriting can differ. Secondly, taking multiple writers into account in case of unconstrained handwriting recognition, a variation in orientation and size of the individual characters is encountered. Therefore, most handwriting recognition systems perform a text line normalization prior to feature extraction in order to standardize the appearance of the handwriting.

In the following, text line normalization is discussed in Section 2.1 and handwriting serialization by means of a sliding window is presented in greater detail in Section 2.2.

2.1. Text line normalization

For text line normalization, the appearance of a text line image in terms of orientation and size is standardized by a series of rotation, shearing, and scaling transformations. The parameters needed for these transformations, e.g., the rotation angle, are typically estimated on a binarized version of the text line image where the text foreground is separated from the document background (Pratikakis et al., 2010; Trier and Taxt, 1995). For historical documents, in particular, binarization is a difficult problem due to paper or parchment degradation artifacts that can result in considerable background noise (Antonacopoulos and Downton, 2007; Su et al., 2010).

If no constraints are imposed on the writing instrument, the stroke width is often normalized in the binary image by means of thinning or skeletonization methods. The aim of these procedures is to obtain a stroke of one pixel width while maintaining the original stroke connectivity (Lam et al., 1992).

The orientation of the baseline, i.e., the skew, often deviates from the horizontal direction and needs to be corrected by rotation. Popular methods for skew angle estimation include contour minima interpolation (Bozinovic and Srihari, 1989) and entropy maximization of horizontal projection profiles (Vinciarelli and Luettin, 2001). Another orientation property of the handwriting that varies from one writer to another is the inclination of the individual characters, i.e., the slant. It can be removed by means of a shear transform. To that end, the mean direction of straight line segments is commonly used as an estimation of the slant (Caesar et al., 1993; Vinciarelli and Luettin, 2000).

For normalizing the size of Roman scripts, the estimated midpoint or x-height of the handwriting, i.e., the height of the lowercase letter “x,” can be used in combination with the baseline for vertical scaling into three disjoint areas of equal height (upper, middle, and lower area) (Marti and Bunke, 2001). Another known practice for normalizing the size of the handwriting is to apply horizontal scaling with respect to the average character width (Saon, 1999).

In Fig. 1, the most commonly applied normalization procedure is illustrated. After correcting the skew, the slant of the handwriting is removed by means of a shear transform. Note that depending on the type of document there might be good reasons to avoid some of the text line normalization methods, since any errors made at this stage may heavily impede the overall recognition process.

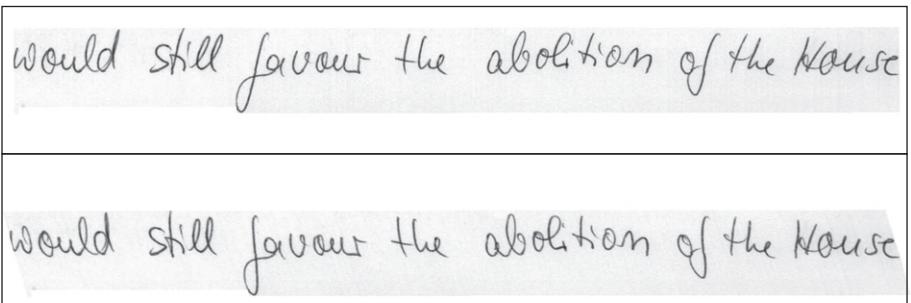


Fig. 1. Slant correction.

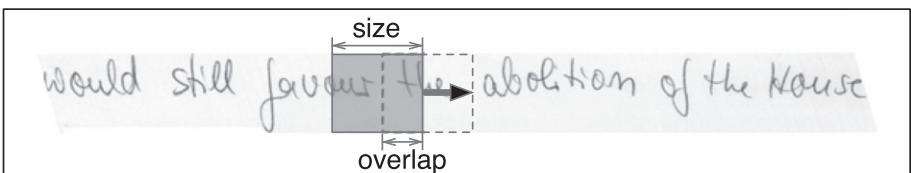


Fig. 2. Sliding window.

2.2. Sliding window feature extraction

For handwriting serialization using a sliding window, an analysis window is moved in writing direction over the text line image, usually from left to right for Roman scripts and from right to left for Arabic scripts (El-Hajj et al., 2005). At each position of the sliding window, a feature vector, which is typically real-valued, is extracted from the subimage captured by the window. Taking into account different window sizes and overlaps, this results in a feature vector sequence description $\mathbf{x} = x_1, \dots, x_T$ of the normalized text line image with $x_i \in \mathbb{R}^n$. An illustration of the sliding window approach is shown in Fig. 2.

It is generally agreed upon the benefit of feature-based representation of handwritten text as opposed to using image pixel intensities directly for recognition. First, a reduction of the amount of data decreases the number of model parameters that need to be estimated on learning samples and secondly, sophisticated features are able to focus on the essential properties of the handwriting that are independent of the capturing device and writing style (Ploetz and Fink, 2009).

Handwriting features are typically extracted from binary or skeletonized versions of the text line image, i.e., after the document background has been removed (see Section 2.1). In order to extract stroke gradients, grayscale information is sometimes added again by smoothing the binary image (Rodriguez and Perronnin, 2008).

In the literature, a large variety of different feature sets have been proposed that can roughly be classified into two categories, i.e., *analytical features* based on low-level pixel analysis and *perceptual features* that are based on a structural analysis of the handwriting (Hammerla et al., 2010). Analytical features include, e.g., contour

position, number of black–white transitions, and average local pixel intensities (Marti and Bunke, 2001; Vinciarelli et al., 2004). In addition, the feature derivatives over the width of the window can capture dynamic aspects as proposed, e.g., by Wienecke et al. (2005).

Perceptual features, on the other hand, capture structural properties such as loops, ascenders, descenders, and concavities (El-Hajj et al., 2005; El Yacoubi et al., 1999). Recently, structural similarity features have been proposed for historical documents by Fischer et al. (2010) with respect to a graph-based representation of handwriting skeletons. Structural features are emerging for Arabic scripts, in particular, where small strokes and dots play an important role for distinguishing different letters. For example, as shown by Saleem et al. (2009), the use of gradient, structure, and concavity features (GSC) in addition to analytical features has proven to be beneficial for HMM-based Arabic handwriting recognition.

In general, it is difficult to decide which feature set is best suited for a given task. However, the quality of some basic features can often be improved by automatic feature selection and dimensionality reduction, e.g., using PCA, ICA, and kernel PCA (Fischer and Bunke, 2009; Vinciarelli and Bengio, 2002).

3. HMM-based text line recognition

HMM-based recognition of text line images aims at finding the most probable sequence of words $\mathbf{w} = w_1, \dots, w_N$ for a given sequence of feature vectors $\mathbf{x} = x_1, \dots, x_T$ obtained by serializing the text line image (see Section 2). For finding the optimal word sequence \mathbf{w} , the posterior probability $P(\mathbf{w}|\mathbf{x})$ is maximized. According to Bayes' rule (Duda et al., 2001), we obtain

$$\mathbf{w} = \underset{\mathbf{w} \in \mathcal{W}}{\operatorname{argmax}} P(\mathbf{w}|\mathbf{x}) = \underset{\mathbf{w} \in \mathcal{W}}{\operatorname{argmax}} \frac{P(\mathbf{x}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x})} \quad (1)$$

based on the likelihood $P(\mathbf{x}|\mathbf{w})$, the prior probability $P(\mathbf{w})$, the evidence $P(\mathbf{x})$, and all possible word sequences \mathcal{W} .²

Note that for handwriting recognition, the majority of the recognition systems are based on real-valued feature vectors and hence continuous HMMs are employed. For continuous HMMs, the likelihood $P(\mathbf{x}|\mathbf{w})$ as well as the evidence $P(\mathbf{x})$ are given by probability density function (pdf) values. In contrast, probabilities used for discrete HMMs are obtained, e.g., by vector quantization (Gersho and Gray, 1991).

In the following, the individual components of the posterior probability are discussed in detail for text line recognition. The likelihood $P(\mathbf{x}|\mathbf{w})$ gives an answer to the question *how* the text line is written with respect to trained character models. This is the most fundamental question that can be answered by using HMMs for handwriting recognition. It is discussed in Section 3.1.

²Unlike earlier publications on HMMs for speech recognition (Rabiner, 1989), a more compact notation is used that hides individual HMM states and model parameters. This notation has become popular in more recent publications (Pitrelli et al., 2006) and puts an emphasis on the language modeling component $P(\mathbf{w})$ at the word-level (see Section 3.2).

The prior probability $P(\mathbf{w})$ is given by a language model and takes into account what is written in the text line. Language modeling has been an emerging topic in the last two decades (Vinciarelli et al., 2004) and the majority of the recognition systems nowadays include basic n -gram language models (Zimmermann and Bunke, 2004), which are introduced in Section 3.2.

Finally, the evidence $P(\mathbf{x})$ reflects *how well* the text line is written *in general* with respect to all possible word sequences \mathcal{W} . While this component is not dependent on \mathbf{w} and thus can be ignored for finding the optimal word sequence, it is necessary for normalizing the recognition score in order to obtain the posterior probability. The posterior probability represents a *confidence measure* of the recognition result that can be used in several ways to improve the performance of a text line recognizer. Confidence modeling is an active area of current research (Pitrelli et al., 2006). Some established methods are discussed in Section 3.3.

3.1. Hidden Markov models

In this section, the HMM topology and parameterization that is typically used for off-line handwriting recognition of text line images is introduced. Based on character HMMs that are trained on labeled text line samples, it is shown how the likelihood $P(\mathbf{x}|\mathbf{w})$ of the feature vector sequence $\mathbf{x} = x_1, \dots, x_T$ can be computed efficiently for a sequence of words $\mathbf{w} = w_1, \dots, w_N$ in order to find the optimal word sequence

$$\mathbf{w} = \underset{\mathbf{w} \in \mathcal{W}}{\operatorname{argmax}} P(\mathbf{x}|\mathbf{w}) \quad (2)$$

taking into account all possible word sequences \mathcal{W} . Here, the result obtained is based on the appearance of the handwriting only, i.e., the correspondence with the trained character models, not taking into account language constraints on the word sequence.

3.1.1. Models

The basic modeling unit is given by a character HMM. It describes two stochastic processes. The first is a first-order Markov chain that produces a series of hidden, i.e., not directly observable, states $\mathbf{s} = s_1, \dots, s_t$ from a finite set S with the Markov property

$$P(s_t|s_1, \dots, s_{t-1}) = P(s_t|s_{t-1}). \quad (3)$$

That is, each state is only dependent on its direct predecessor. This process can be represented by a finite state automaton as shown in Fig. 3 for the linear topology. Given a set of states $S = \{S_1, \dots, S_m\}$ and starting in state S_1 with an initial state probability of $\pi_1 = P(S_1) = 1$, the model either rests in a state or changes to the next state with transition probabilities $a_{i,i} = P(S_i|S_i)$ and $a_{i,i+1} = P(S_{i+1}|S_i)$ such that $a_{i,i} + a_{i,i+1} = 1$. The linear topology shown in Fig. 3, instead of an ergodic one where each state can be a successor to any other state, is typical for handwriting recognition. Here, the first states represent the beginning of a character, followed by the middle and end parts. Sometimes, a Bakis topology is preferred, where the skipping of one state is possible (Ploetz and Fink, 2009).

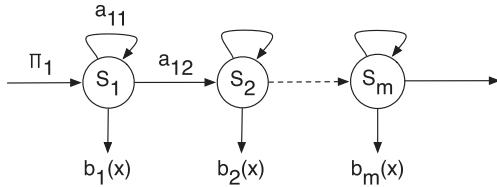


Fig. 3. Character HMM.

The second stochastic process that is described by the character HMM produces a series of observable feature vectors $\mathbf{x} = x_1, \dots, x_t$ with typically continuous features $x_t \in \mathbb{R}^n$. Such an output is emitted at every time t by the current state $s_t = S_i$ and is commonly modeled by a mixture of Gaussians

$$b_i(x) = \sum_{k=1}^G w_{ik} \mathcal{N}(x | \mu_{ik}, \Sigma_{ik}), \quad (4)$$

where G is the number of Gaussians, $\mathcal{N}(x | \mu_{ik}, \Sigma_{ik})$ is a normal distribution with mean μ_{ik} and covariance matrix Σ_{ik} , and w_{ik} is the prior probability of the k th mixture. In order to reduce the number of parameters that need to be estimated on learning samples, diagonal covariance matrices are frequently considered. Another technique involves parameter tying for so-called semicontinuous HMMs, where a shared set of Gaussians is used for all states. Both techniques were used, e.g., by Wienecke et al. (2005) for camera-based whiteboard reading. Here, a total of 75 characters are modeled with semicontinuous HMMs that share a codebook of 1500 Gaussians with diagonal covariance matrices.

Given a set of m states $S = \{S_1, \dots, S_m\}$, the model parameters of a character HMM are, in summary, given by

$$\lambda = (\pi, A, B), \quad (5)$$

where $\pi = \{\pi_i\}$ are the initial state probabilities, $A = \{a_{ij}\}$ the transition probabilities, and $B = \{b_i(x)\}$ the emission pdfs for $1 \leq i, j \leq m$.

3.1.2. Algorithms

An important reason for the widespread use of HMMs for handwriting recognition is given by the availability of efficient algorithms for training and recognition. The centerpiece of these algorithms is the total or partial calculation of the observation pdf $P(\mathbf{x}|\lambda)$ of the observed feature vector sequence $\mathbf{x} = x_1, \dots, x_T$ for a set of HMM parameters λ given by

$$P(\mathbf{x}|\lambda) = \sum_{\mathbf{s} \in \mathcal{S}} P(\mathbf{x}|\mathbf{s}, \lambda) P(\mathbf{s}|\lambda) \quad (6)$$

with respect to all possible state sequences $\mathbf{s} = s_1, \dots, s_T \in \mathcal{S}$, the pdf $P(\mathbf{x}|\mathbf{s}, \lambda)$ of a given state sequence, and its prior probability $P(\mathbf{s}|\lambda)$. The state sequence specific terms are given by

$$P(\mathbf{s}|\lambda) = \pi_{s_1} \prod_{t=2}^T a_{s_{t-1}, s_t} \quad (7)$$

$$P(\mathbf{x}|\mathbf{s}, \lambda) = \prod_{t=1}^T b_{s_t}(x_t) \quad (8)$$

assuming statistical independence of the observation sequence \mathbf{x} in Eq. (8). This assumption is not really valid for handwriting recognition, since the observations are usually highly correlated. This is an area for potential future improvements. However, the estimation of a more complicated representation might not provide an overall better performance (Juang and Rabiner, 1991).³ In practice, logarithmic values are typically considered in Eqs. (7) and (8) in order to make computations with very small values feasible.

A straight-forward calculation of $P(\mathbf{x}|\lambda)$ in Eq. (6) over all possible state sequences $\mathbf{s} \in \mathcal{S}$ would result in an exponential complexity of $O(N^T T)$ in terms of the number of states N and the observation sequence length T , which is not feasible in practice. Fortunately, the complexity can be reduced by means of dynamic programming based on the lattice representation shown in Fig. 4 exemplarily for the linear topology.⁴ Based on this lattice, the computation of $P(\mathbf{x}|\lambda)$ can be done in $O(N^2 T)$ time using the Forward–Backward algorithm (Baum and Egon, 1967). By this procedure, $P(\mathbf{x}|\lambda)$ is accumulated iteratively in the lattice over time, taking only a maximum of N parent nodes from time $t - 1$ (forward) or $t + 1$ (backward) into account for each node.

For training the character HMMs, a text line HMM is created first as a concatenation of character models, including the special space character “sp,” according to the correct transcription of a given training sample as shown in Fig. 5 for the transcription “text line.” Then, the conjoint text line model parameters λ are optimized with respect to $P(\mathbf{x}|\lambda)$ from Eq. (6), typically using the Baum–Welch algorithm (Rabiner, 1989). This variant of the Expectation Maximization (EM) algorithm (Dempster et al., 1977) iteratively adapts the model parameters in order to increase $P(\mathbf{x}|\lambda)$ until the improvement becomes marginal. Note that for continuous HMMs the Baum–Welch algorithm needs a few adaptations when compared to discrete HMMs (Juang et al., 1986). A great advantage of this training approach is that the individual character HMMs are trained conjointly and thus, no segmentation of the text line into character images is needed.

³An interesting comment on parameter significance mentioned by Juang and Rabiner (1991, p. 269) in the context of speech recognition concerns the significance of the transition probabilities a_{ij} when compared with the emission pdfs b_i . Due to the almost unlimited dynamic range of the pdfs, the b_i values tend to dominate the transition probabilities, such that the same system performance can be achieved in practice by neglecting the transition probabilities entirely, i.e., by assuming $a_{i,i} = a_{i,i+1} = 0.5$ for linear topologies. For the estimation of the emission pdfs, however, the transition probabilities still play an important role.

⁴The term *lattice* is used for the planar graph representation of the dynamic programming approach. The graph is ordered along the x -axis by time and is sometimes also called *trellis*.

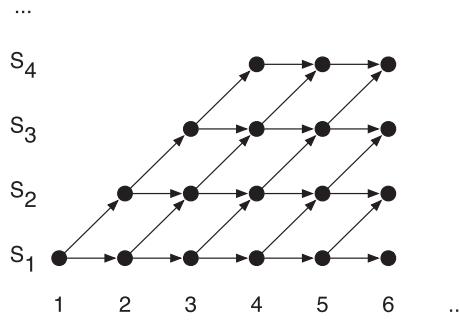


Fig. 4. Recognition lattice.

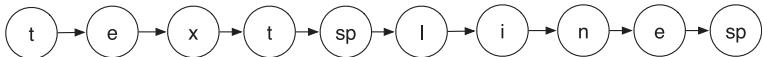


Fig. 5. Training text line HMM.

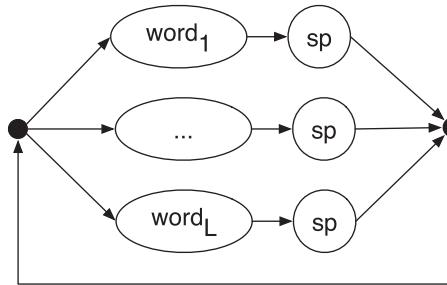


Fig. 6. Recognition text line HMM.

For HMM-based text line recognition, the trained character HMMs are concatenated to word HMMs of a given lexicon \mathcal{L} of size $|\mathcal{L}| = L$. The text line can then be modeled as a large HMM containing a loop over all words that are separated from each other by the space character “sp” as illustrated in Fig. 6. The optimal sequence of states $\mathbf{s} = s_1, \dots, s_T$ can then be found with respect to $P(\mathbf{x}, \mathbf{s} | \lambda)$ by means of the Viterbi algorithm (Rabiner, 1989). Based on dynamic programming using the lattice representation illustrated in Fig. 4, the score

$$\phi_t(S_j) = \max_{1 \leq i \leq N} (\phi_{t-1}(S_i) a_{ij}) b_j(x_t) \quad (9)$$

is assigned to each state S_j at time t taking into account Eqs. (7) and (8). Based on the initialization

$$\phi_1(S_j) = \pi_j b_j(x_1) \quad (10)$$

for all states S_j , the maximum observation pdf

$$P(\mathbf{x}, \mathbf{s} | \lambda) = P(\mathbf{x} | \mathbf{s}, \lambda) P(\mathbf{s} | \lambda) = \max_{1 \leq j \leq N} \phi_T(S_j) \quad (11)$$

is returned alongside with the optimal sequence of states \mathbf{s} . From the optimal sequence of states, the optimal sequence of words \mathbf{w} is finally retrieved based on the word start and end states and provides a solution to Eq. (2). The computational time needed for Viterbi recognition of the text line is $O(L^2T)$ with respect to the lexicon size L and the length of the observation sequence T . In order to speed up the process for large lexicons, pruning is often performed, e.g., by means of beam search (Koerich et al., 2003).

Variants of the text line recognition approach include the special case of single word recognition, where no loop over the words is needed, and lexicon-free recognition, where the basic modeling units are characters rather than words and an optimal sequence of characters is returned. The latter has been successfully used, e.g., by Schambach (2008) for postal address reading in combination with a post-processing step to extract valid words from the output.

The great advantage of HMM-based text line recognition is given by the fact that no segmentation of the text line into words or characters is needed. The segmentation is performed implicitly during recognition, while finding the most likely sequence of states, and is returned as a byproduct of the recognition. The word boundaries that are returned by the Viterbi algorithm can be used to perform a segmentation of text a line image into words, which is useful, e.g., for transcription alignment tasks (Fischer et al., 2011; Indermühle et al., 2009).

3.1.3. Recognition performance

The performance of a text line recognition system is measured by the word accuracy. Hereby, the computed optimal word sequence \mathbf{w} is aligned with the ground truth of test samples by means of string edit distance (Wagner and Fischer, 1974). The word accuracy A is then given by

$$A = \frac{N - S - D - I}{N},$$

where N is the number of words in the ground truth, S is the number of word substitutions, D is the number of deletions, and I is the number of insertions. Note that for a high number of word insertions, the word accuracy can become negative.

Important HMM parameters, which greatly affect the word accuracy and thus need to be carefully fine-tuned on a validation set in practice, are the number of Gaussian mixtures used for the emission pdf and the number of states for each character. This quantity typically depends on the estimated mean character width (Zimmermann and Bunke, 2002).

3.2. Language models

Based on the text line recognition approach discussed in Section 3.1, the optimal sequence of words $\mathbf{w} = w_1, \dots, w_N$ is found with respect to the maximum likelihood $P(\mathbf{x} | \mathbf{w})$ of the feature vector sequence \mathbf{x} , i.e., based on the correspondence of the

trained character HMMs with the image. The implicit assumption of this approach is that all word sequences have the same a priori probability $P(\mathbf{w})$. Of course, this assumption often does not hold true, e.g., for natural language, where words do not occur arbitrarily in a text. Language models aim at capturing the constraints that are imposed on word sequences in a given language and can be used to improve the maximum posterior probability estimation from Eq. (1) with

$$\mathbf{w} = \underset{\mathbf{w} \in \mathcal{W}}{\operatorname{argmax}} P(\mathbf{x}|\mathbf{w})P(\mathbf{w}) \quad (12)$$

taking into account non-trivial prior probabilities $P(\mathbf{w})$ in addition to the HMM likelihood $P(\mathbf{x}|\mathbf{w})$. Including language models for HMM-based recognition can be regarded as a standard approach nowadays (Ploetz and Fink, 2009).

3.2.1. *n*-Gram models

The most frequently used approach to language modeling for handwriting recognition is based on *n*-grams. Here, the underlying stochastic process that produces the sequence of words $\mathbf{w} = w_1, \dots, w_t$ is a Markov chain of order $m = n - 1$ with the property

$$P(w_t|w_1, \dots, w_{t-1}) = P(w_t|w_{t-m}, \dots, w_{t-1}). \quad (13)$$

That is, each word is only dependent on its m predecessors that are also referred to as the history of w_t . The corresponding probabilities of the unigrams $P(w_t)$, bigrams $P(w_t|w_{t-1})$, trigrams $P(w_t|w_{t-2}, w_{t-1})$, and so on have to be estimated on a training corpus of texts written in the language under consideration. Examples of large publicly available corpora include, e.g., the *Brown Corpus of Standard American English* (Francis and Kucera, 1979) and the *Lancaster-Oslo/Bergen Corpus* (LOB) (Johansson et al., 1978) for British English.

In practice, bigrams are the predominant type of language model for HMM-based handwriting recognition (Vinciarelli et al., 2004). The reason is twofold. First, the additional gain in terms of word accuracy reported for *n*-grams of higher order, e.g., trigrams, is typically rather low (Zimmermann and Bunke, 2004). Secondly, larger text corpora are needed for a reliable estimation of high order *n*-grams. For bigrams, the word sequence probability is given by

$$P(\mathbf{w}) = P(w_1) \prod_{i=2}^t P(w_i|w_{i-1}) \quad (14)$$

with the unigram probability $P(w_1)$ and the bigram probabilities $P(w_i|w_{i-1})$. In principle, the *n*-gram probabilities can simply be estimated by their relative occurrence in the training corpus. However, zero probabilities are an unrealistic estimate in case of *n*-grams that do not occur in the training corpus at all. Such an unseen *n*-gram would result in a zero probability of the complete word sequence. Therefore, several smoothing techniques have been proposed to deal with unseen *n*-grams, e.g., based on backing-off and interpolation (Katz, 1987; Kneser and Ney, 1995).

The bigram language model can be integrated into HMM-based recognition by modifying the score used for Viterbi recognition in Eq. (9). With respect to logarithmic values and the bigram probability $w_{ij} = P(w_j|w_i)$, the modified score is given by

$$\phi_t(S_j) = \max_{1 \leq i \leq N} (\phi_{t-1}(S_i) + \log a_{ij} + \alpha \log w_{ij} + \beta) + \log b_j(x_t) \quad (15)$$

whenever the state change from S_i to S_j is a word change. Hereby, the parameters α and β are related to the integration of the language model. The factor $\alpha \geq 0$ is called *grammar scale factor* and weights the impact of the bigram word model against the likelihood of the feature vector sequence. The factor $\beta \in \mathbb{R}$ is called *word insertion penalty* and balances the number of word insertions during recognition. Both factors have a great influence on the overall word accuracy and thus need to be fine-tuned on a validation set in practice.

3.2.2. Language model performance

In order to evaluate the quality of a language model independently of the recognition, a frequently used criterion is given by the perplexity

$$\mathcal{P}(\mathbf{w}) = P(\mathbf{w})^{-\frac{1}{T}} \quad (16)$$

of an unseen word sequence $\mathbf{w} = w_1, \dots, w_T$ that was not used for creating the language model. The perplexity reflects the cross-entropy between the language model and the word distribution in the unseen data (Jelinek et al., 1982). In the worst case, each word occurs with the same probability $P(w_t) = \frac{1}{L}$ independently of its context. In this case, for lexicon size L , the resulting perplexity is L . If the language model can better predict the word sequence, a lower perplexity $1 \leq \mathcal{P}(\mathbf{w}) \leq L$ is obtained. Intuitively, the perplexity can be regarded as the lexicon reduction that is achieved by the language model.

3.3. Confidence models

In Section 3.2, it was shown how the optimal word sequence \mathbf{w} can be found for a given feature vector sequence \mathbf{x} with respect to $P(\mathbf{x}|\mathbf{w})P(\mathbf{w})$, taking into account the appearance likelihood $P(\mathbf{x}|\mathbf{w})$ as well as the language model probability $P(\mathbf{w})$. In practice, the word accuracy that can be achieved with this word sequence model is regarded nowadays as still being far from perfect, although strong progress could be observed in the past two decades (Vinciarelli et al., 2004). This progress can be illustrated for the IAM database (Marti and Bunke, 2002), which provides an established benchmark dataset for unconstrained off-line cursive handwriting recognition. Over 600 writers have contributed to this dataset that consists of over 13,000 handwritten text lines from the LOB corpus (Johansson et al., 1978) for British English. Shortly after the introduction of the IAM database 10 years ago, a word accuracy of about 60% was reported for HMM-based recognition using bigram language models (Marti and Bunke, 2001). Meanwhile, the best reported word accuracy for the IAM database, to the knowledge of the authors, is about 74%

with a hybrid HMM/NN approach that employs neural networks (NN) to compute the HMM emission pdfs (Espana-Boquera et al., 2011).

In order to improve the reliability of handwriting recognition systems for real-world applications, a common approach is to endow the recognition result \mathbf{w} with a confidence $C(\mathbf{w})$, preferably with values between 0 and 1. By comparing the confidence score

$$C(\mathbf{w}) \geq T \quad (17)$$

with a threshold T , unreliable results with $C(\mathbf{w}) < T$ can be rejected in order to process them manually afterward, e.g., in postal address reading (Brakensiek et al., 2003; El Yacoubi et al., 2002). There are many other application areas that are dependent on a confidence score, e.g., writer identification (Schlapbach and Bunke, 2004), writer verification (Schlapbach and Bunke, 2006), combination of multiple classifier systems (Bertolami and Bunke, 2008; Indermühle et al., 2009), interactive transcription (Tarazón et al., 2009; Toselli et al., 2010), and keyword spotting (Edwards et al., 2004; El Yacoubi et al., 2002; Fischer et al., 2012; Rodriguez and Perronnin, 2009; Thomas et al., 2010).

Several confidence measures were proposed in the handwriting recognition literature. They can be broadly classified into the three categories *heuristic*, *posterior based*, and *likelihood ratio based*. The first includes, e.g., likelihood stability under variation of language model parameters (Bertolami and Bunke, 2008) and character length stability for different handwriting recognition systems (Indermühle et al., 2009). In this chapter, we discuss the two more general approaches in greater detail, i.e., posterior based and likelihood ratio based.

From a Bayesian point of view, the posterior probability is a natural measure of confidence that can be used to find an optimum tradeoff between false acceptance and false rejection of recognition results with respect to the user's preference (Chow, 1970). For handwriting recognition, the posterior-based confidence is given by

$$C(\mathbf{w}) = P(\mathbf{w}|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x})}, \quad (18)$$

where the evidence $P(\mathbf{x})$ is needed in addition to the terms discussed so far. The computation of this confidence measure is discussed in Section 3.3.1. By comparing the posterior of the optimal solution with the posterior of a competing alternative, the frequently considered likelihood ratio-based confidence measure can be obtained. This measure is able to avoid the costly computation of $P(\mathbf{x})$. It is presented in Section 3.3.2. Finally, the Receiver Operating Characteristic (ROC) curve is introduced in Section 3.3.3 as a widely used performance measure for confidence-based recognition. With respect to this performance measure, the optimal confidence threshold from Eq. (17) can be determined based on the user's preference (Tortorella, 2000).

3.3.1. Posterior-based confidence

For the posterior probability $P(\mathbf{w}|\mathbf{x})$ of the optimal sequence of words \mathbf{w} , the evidence $P(\mathbf{x})$, sometimes also referred to as *score normalization*, is given by

$$P(\mathbf{x}) = \sum_{\mathbf{w} \in \mathcal{W}} P(\mathbf{x}|\mathbf{w})P(\mathbf{w}). \quad (19)$$

It is based on all possible word sequences \mathcal{W} and can be obtained efficiently by means of the Forward–Backward algorithm in $O(L^2T)$ time with respect to the size L of the lexicon and the size T of the observation sequence \mathbf{x} (see Section 3.1.2).⁵ In practice, however, the computational complexity is still challenging in real-world applications with very large lexicons. Hence, approximate solutions are considered that can be found, e.g., by means of a beam search that takes only a limited number of word sequences with a high likelihood into account.

For text line recognition, a direct implementation of Eq. (19) based on word graphs was recently presented by Tarazón et al. (2009) for an interactive transcription system. Originally proposed for speech recognition by Wessel et al. (2001), each word sequence $\mathbf{w} \in \mathcal{W}$ is represented in a graph similar to the state-based recognition lattice discussed in Section 3.1.2. The word graph can be obtained as a byproduct of the Viterbi decoding by recording possible word endings at each time step. The nodes of the graph represent discrete points in time and the arcs are labeled with a word, its start and end position, and its likelihood. Using this word graph, the posterior probability of the text line or the posterior probability of a single word within the text line is then calculated by means of the Forward–Backward algorithm. The computational complexity of the procedure can be reduced by considering only the N -best words at each time step for creating the word graph.

For the special case of single word recognition, the observation sequence \mathbf{x} of a word image can be presented to all words $w \in \mathcal{L}$ of the lexicon \mathcal{L} and the evidence is simplified to

$$P(\mathbf{x}) = \sum_{w \in \mathcal{L}} P(\mathbf{x}|w)P(w) \quad (20)$$

taking into account the unigram probability $P(w)$.⁶ In the literature, the resulting posterior $P(w|\mathbf{x})$ has been used, e.g., by Koerich (2004) for word rejection and by Pitrelli et al. (2006) in the general context of confidence modeling for on-line handwriting recognition.

If no lexicon is available or the computational cost of computing lexicon-based posteriors is too high, an approximation of $P(\mathbf{x})$ by means of so-called *filler* models can be considered. Also known as *garbage* or *word* models, they represent general text content disregarding lexicon and language model constraints. The evidence is then approximated by

$$P(\mathbf{x}) \sim P(\mathbf{x}|\mathbf{f})P(\mathbf{f}) \quad (21)$$

with respect to the optimal sequence of filler models \mathbf{f} . Typical filler models are given by character HMMs, sometimes endowed with character bigrams for non-trivial

⁵In contrast to HMM training, only the forward part of the algorithm is needed for calculating the evidence.

⁶For single word recognition, a uniform distribution of the words over the lexicon is often assumed. In this case, the recognition is based on the text appearance only.

priors $P(\mathbf{f})$ (Brakensiek et al., 2003; Edwards et al., 2004; Fischer et al., 2012; Thomas et al., 2010). With this approach, the computational complexity of calculating the evidence is drastically reduced to $O(A^2 T)$ with respect to the alphabet size A . The posterior probability of single words within a text line can be obtained by taking into account the local filler likelihood between a word's starting and ending position. The filler-based score normalization can either be integrated into the recognition process (Edwards et al., 2004; Thomas et al., 2010) or be added as a post-processing module (Brakensiek et al., 2003; Fischer et al., 2012). As a variant of character HMMs, a single GMM, referred to as *universal vocabulary*, has recently been proposed by Rodriguez and Perronnin (2009) for score normalization in the context of keyword spotting.

3.3.2. Likelihood ratio-based confidence

Instead of using the posterior probability $P(\mathbf{w}|\mathbf{x})$ directly as a confidence measure (see Section 3.3.1), there are a number of approaches known from the literature that obtain the confidence from the closely related posterior ratio between the best word sequence \mathbf{w}_{1st} and the second best word sequence \mathbf{w}_{2nd} , as reported, e.g., by Brakensiek et al. (2003), Koerich (2004), Marukatat et al. (2002), Pitrelli et al. (2006)). The corresponding confidence measure is then given by

$$C(\mathbf{w}_{1st}) = \frac{P(\mathbf{w}_{1st}|\mathbf{x})}{P(\mathbf{w}_{2nd}|\mathbf{x})} = \frac{P(\mathbf{x}|\mathbf{w}_{1st})P(\mathbf{w}_{1st})}{P(\mathbf{x}|\mathbf{w}_{2nd})P(\mathbf{w}_{2nd})} \quad (22)$$

with respect to the likelihood ratio. An advantage of the likelihood ratio-based confidence measure is that the costly computation of the evidence $P(\mathbf{x})$ is not needed. In the report of Marukatat et al. (2002), the likelihood ratio-based confidence is investigated in the context of on-line text line recognition. Based on a word graph, obtained by Viterbi decoding with a beam search, the likelihood ratio to the second best word hypothesis is used, among other approaches, as a confidence measure for individual words within the text line. For the special case of single word recognition, the same confidence modeling approach is pursued, e.g., by Brakensiek et al. (2003), Koerich (2004) and Pitrelli et al. (2006).

Another approach to likelihood ratio-based confidence measures is given by statistical hypothesis testing. According to Neyman-Pearson's Lemma (Neyman and Pearson, 1933), the likelihood ratio $\frac{f(x|\theta_0)}{f(x|\theta_1)}$ with respect to a probability density function $f(x|\theta)$ with parameter θ provides an optimal test for the null hypothesis $H_0 : \theta = \theta_0$ and alternative hypothesis $H_1 : \theta = \theta_1$. For HMM-based handwriting recognition, this test can be stated as

$$C(H_0) = \frac{P(\mathbf{x}|\lambda_0)}{P(\mathbf{x}|\lambda_1)} \quad (23)$$

with respect to the observation likelihood from Eq. (6) based on the HMM parameters λ_0 and λ_1 of the null and alternative hypotheses, respectively. In practice, an approximation of Eq. (23) is used based on the likelihood of the most probable sequence of HMM states that can be obtained by Viterbi decoding. Another approximation is given by the fact that the true probability density function is unknown and has to be estimated from the training data assuming a Gaussian

mixture model. Hence, the test is not optimal, but has proven to be very successful, in particular in the field of speech recognition (Rose et al., 1995).

When the optimal word sequence $\mathbf{w}_{1\text{st}}$ is considered as the null hypothesis and the second best word sequence $\mathbf{w}_{2\text{nd}}$ as the alternative hypothesis, Eq. (23) is, in fact, approximated by Eq. (22). In the handwriting recognition literature, there are a number of reports that use different alternative models, often referred to as *anti-models* or *cohort models*, to obtain a confidence measure. In the report of Schlapbach and Bunke (2004), different writer-specific HMMs are trained for the task of writer identification and the likelihood ratio between the best writer model and the second best is used as a confidence measure. An additional writer-independent anti-model was used by Schlapbach and Bunke (2006) for confidence modeling in the context of writer verification.

Also, the filler models discussed in Section 3.3.1 have been used frequently as general anti-models for text, e.g., by Brakensiek et al. (2003), Koerich (2004), Marukatat et al. (2002), Pitrelli et al. (2006). In fact, the same confidence measure

$$C(\mathbf{w}) = \frac{P(\mathbf{x}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x}|\mathbf{f})P(\mathbf{f})} \quad (24)$$

results for the likelihood ratio as for the posterior approximation taking into account Eqs. (18) and (21).

In general, it is difficult to predict which confidence model will perform best for a given task, considering different constraints on available training data and computation time. Instead, the best model is typically chosen on a validation set with respect to the system's performance. This issue will be discussed in the next section. For off-line handwritten text line recognition, a comprehensive experimental evaluation and comparison of different confidence models is still missing to the knowledge of the authors.

3.3.3. Confidence model performance

The performance of a confidence-based recognition system is measured by its ability to identify and reject unreliable recognition results. Given a confidence value, the result is accepted only if the value exceeds a certain threshold and is rejected otherwise. For evaluation, the numbers of correct acceptance (CA), false acceptance (FA), correct rejection (CR), and false rejection (FR) are used to calculate the false acceptance rate (FAR) and false rejection rate (FRR) given by

$$\text{FAR} = \frac{\text{FA}}{\text{FA} + \text{CR}}, \quad (25)$$

$$\text{FRR} = \frac{\text{FR}}{\text{FR} + \text{CA}}. \quad (26)$$

These two quantities define the Receiver Operating Characteristic (ROC) curve (Tortorella, 2000). Based on different thresholds, a trade-off between FAR and FRR can be achieved that is suitable for a given application. For example, in a postal address reading application, it might be necessary to achieve a very small false acceptance rate in order to prevent wrong deliveries, at the cost of a higher false rejection rate that leads to an increased amount of manual post-processing.

4. Outlook and conclusions

For the challenging task of handwriting recognition in document images, HMMs are probably the most widely used statistical models nowadays. In this chapter, we reviewed state-of-the-art methods for segmentation-free recognition of handwritten text lines for alphabetical languages, large vocabularies, and multiple writers. These methods include the serialization of text line images with a sliding window (Section 2), character HMM modeling with Gaussian mixture emissions (Section 3.1.1), efficient Baum–Welch training and Viterbi decoding for finding the optimal sequence of words (Section 3.1.2), and language modeling with word n -grams (Section 3.2). Besides these well-established methods, an introduction to confidence modeling is provided, which is an active area of current research (Section 3.3).

Without doubt, the application of HMMs has lead to a strong progress in the field of handwriting recognition in the past two decades. Still, the ultimate goal of a universal reading machine for handwriting that can compete with the accuracy of a human reader (Lorette, 1999) is still far from being reached. There are, however, many areas for improvement. One of the major challenges is given by the fact that current n -gram language modeling approaches are still rather primitive when compared to a true *understanding* of the handwriting that allows humans to recognize even badly written words by considering the general context. Only very few attempts have been made so far to extend the predominant n -gram based language modeling approach, e.g., by integrating stochastic context-free grammars (SCFG) into HMM-based recognition based on grammatical word tags (e.g., noun, pronoun, verb form) (Zimmermann et al., 2006).

Another important challenge is to cope with uncertainty in recognition. Since fully automatic recognition systems are not yet accurate enough for an unconstrained scenario with large vocabularies and multiple writers, manual interaction is needed in order to obtain error-free results. For such interactive systems, confidence modeling is of crucial importance in order to separate reliable results from unreliable ones and hence reduce the costly effort of manual correction (Tarazón et al., 2009). In this context, there is a recent trend to incorporate multimodal interactions and to improve the recognition system based on the user feedback as proposed, e.g., by Toselli et al. (2010) for HMM-based interactive transcription. Confidence modeling is also needed for other tasks of current interest, such as keyword spotting in historical documents that aim at making document images amenable to browsing and searching in digital libraries (Antonacopoulos and Downton, 2007). HMM-based keyword spotting systems that use filler models for confidence modeling have recently been proposed by Fischer et al. (2012) and Thomas et al. (2010).

Furthermore, the question how to cope with a limited number of training samples is an ongoing issue of current research. Basically, a robust estimation of the various parameters in an HMM-based recognition system is not possible if there is a lack of training data. Because of the large effort needed to manually label a large number of training samples, the generation of synthetic data has been pursued, e.g., by Varga and Bunke (2004). More recently, there is an increasing interest in using unlabeled data, which is cheaply available, for improving the handwriting recognition performance by means of semisupervised learning (Ball and Srihari, 2009).

Finally note that besides the predominant HMM approach to handwriting recognition, there are other competing approaches. Most prominently, bidirectional long short-term memory (BLSTM) neural networks (Graves et al., 2009) have proven to outperform HMMs in several tasks (Grosicki and Abed, 2009). Nevertheless, a great advantage of HMM-based systems is that they are, unlike neural networks, generative systems based on solid statistical principles.

Acknowledgment

This work has been supported by the Swiss National Science Foundation (Project CRSI22-125220).

References

- Antonacopoulos, A., Downton, A., 2007. Special issue on the analysis of historical documents. *Int. J. Doc. Anal. Recogn.* 9, 75–77.
- Ball, G.R., Srihari, S.N., 2009. Semi-supervised learning for handwriting recognition. In: Proceedings of 10th International Conference on Document Analysis and Recognition, pp. 26–30.
- Baum, L., Egon, J., 1967. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bull. Am. Math. Soc.* 73, 360–363.
- Bertolami, R., Bunke, H., 2008. Hidden Markov model based ensemble methods for offline handwritten text line recognition. *Pattern Recognit.* 41, 3452–3460.
- Boccignone, G., Chianese, A., Cordella, L.P., Marcelli, A., 1993. Recovering dynamic information from static handwriting. *Pattern Recognit.* 26, 409–418.
- Bozinovic, R.M., Srihari, S.N., 1989. Off-line cursive script word recognition. *IEEE Trans. PAMI* 11, 68–83.
- Brakensiek, A., Rigoll, G. 2004. Handwritten address recognition using hidden markov models. In: Dengel, A., Junker, M., Weisbecker, A. (Eds.), *Reading and Learning*, vol. 2956, LNCS. Springer, pp. 103–122.
- Brakensiek, A., Rottland, J., Rigoll, G., 2003. Confidence measures for an address reading system. In: Proceedings of 7th International Conference on Document Analysis and Recognition, pp. 294–298.
- Bunke, H., Varga, T., 2007. Off-line Roman cursive handwriting recognition. In: Chaudhuri, B. (Ed.), *Digital Document Processing: Major Directions and Recent Advances*. Springer, vol. 20, pp. 165–173.
- Bunke, H., Ammann, R., Kaufmann, G., Ha, T., Schenkel, M., Seiler, R., Eggimann, F., 1997. Recovery of temporal information of cursively handwritten words for on-line recognition. In: Proceedings of 4th International Conference on Document Analysis and Recognition, pp. 931–935.
- Caesar, T., Gloer, J., Mandler, E., 1993. Preprocessing and feature extraction for a handwriting recognition system. In: Proceedings of 2nd International Conference on Document Analysis and Recognition, pp. 408–411.
- Chow, C., 1970. On optimum recognition error and reject tradeoff. *IEEE Trans. Inform. Theory* 16, 41–46.
- Dempster, A.P., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc.* 39, 1–38.
- Doermann, D., Rosenfeld, A., 1992. Recovery of temporal information from static images of handwriting. In: Proceedings of International Conference on Computer Vision and, Pattern Recognition, pp. 162–168.
- Duda, R.O., Hart, P., Stork, D.G., 2001. *Pattern Classification*. Wiley-Interscience.
- Edwards, J., Teh, Y.W., Forsyth, D., Bock, R., Maire, M., Vesom, G., 2004. Making Latin manuscripts searchable using gHMM's. In: *Advances in Neural Information Processing Systems*, pp. 385–392.

- El-Hajj, R., Likforman-Sulem, L., Mokbel, C. 2005. Arabic handwriting recognition using baseline dependant features and hidden Markov modeling. In: Proceedings of 8th International Conference on Document Analysis and Recognition, pp. 893–897.
- El Yacoubi, A., Gilloux, M., Sabourin, R., Suen, C., 1999. An HMM-based approach for off-line unconstrained handwritten word modeling and recognition. IEEE Trans. PAMI 21, 752–760.
- El Yacoubi, A., Gilloux, M., Bertille, J.M., 2002. A statistical approach for phrase location and recognition within a text line: an application to street name recognition. IEEE Trans. PAMI 24, 172–188.
- Espana-Boquera, S., Castro-Bleda, M., Gorbe-Moya, J., Zamora-Martinez, F., 2011. Improving offline handwritten text recognition with hybrid HMM/ANN models. IEEE Trans. PAMI 33, 767–779.
- Fischer, A., Bunke, H., 2009. Kernel PCA for HMM-based cursive handwriting recognition. In: Proceedings of 13th International Conference on Computer Analysis of Images and Patterns. Springer, pp. 181–188.
- Fischer, A., Riesen, K., Bunke, H. 2010. Graph similarity features for HMM-based handwriting recognition in historical documents. In: Proceedings of 12th International Conference on Frontiers in Handwriting Recognition, pp. 253–258.
- Fischer, A., Indermühle, E., Frinken, V., Bunke, H., 2011. HMM-based alignment of inaccurate transcriptions for historical documents. In: Proceedings of 11th International Conference on Document Analysis and Recognition, pp. 53–57.
- Fischer, A., Keller, A., Frinken, V., Bunke, H., 2012. Lexicon-free handwritten word spotting using character HMMs. Pattern Recogn. Lett. 33, 934–942.
- Francis, W., Kucera, H. 1979. Brown Corpus Manual. Manual of Information to Accompany A Standard Corpus of Present-Day Edited American English, for Use with Digital Computers. Brown University.
- Gatos, B., Stamatopoulos, N., Louloudis, G. 2010. ICFHR 2010 handwriting segmentation contest. In: Proceedings of 12th International Conference on Frontiers in Handwriting Recognition, pp. 737–742.
- Gersho, A., Gray, R.M. 1991. Vector Quantization and Signal Compression. Kluwer Academic Publishers.
- Gorski, N., Anisimov, V., Augustin, E., Baret, O., Maximor, S., 2001. Industrial bank check processing: the A2iA check reader. Int. J. Doc. Anal. Recogn. 3, 196–206.
- Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., Schmidhuber, J., 2009. A novel connectionist system for improved unconstrained handwriting recognition. IEEE Trans. PAMI 31, 855–868.
- Grosicki, E., Abed, H.E., 2009. Icdar handwriting recognition competition. In: Proceedings of 10th International Conference on Document Analysis and Recognition, pp. 1398–1402.
- Hammerla, N.Y., Plötz, T., Vajda, S., Fink, G.A. 2010. Towards feature learning for HMM-based offline handwriting recognition. In: Proceeding of International Workshop on Frontiers in Arabic Handwriting Recognition, pp. 27–32.
- Impedovo, S., Wang, P., Bunke, H. (Eds.), 1997. Automatic Bankcheck Processing. World Scientific.
- Indermühle, E., Liwicki, M., Bunke, H. 2009. Combining alignment results for historical handwritten document analysis. In: Proceedings of 10th International Conference on Document Analysis and Recognition, pp. 1186–1190.
- Jelinek, F., Mercer, R., Bahl, L. 1982. Continuous speech recognition: statistical methods. In: Krishnaiah, P., Kanal, L. (Eds.), Classification Pattern Recognition and Reduction of Dimensionality. Handbook of Statistics. Elsevier, vol. 2. pp. 549–573.
- Johansson, S., Leech, G., Goodluck, H., 1978. Manual of Information to Accompany the Lancaster-Oslo/Bergen Corpus of British English, for Use with Digital Computers. University of Oslo, Department of English.
- Juang, B.H., Rabiner, L.R., 1991. Hidden markov models for speech recognition. Technometrics 33, 251–272.
- Juang, B.H., Levinson, S.E., Sondhi, M.M. 1986. Maximum likelihood estimation for multivariate mixture observations of Markov chains. IEEE Trans. Inform. Theory IT-32, 307–309.
- Katz, S.M., 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. IEEE Trans. Acoust. Speech Signal Process. 35, 400–401.
- Kneser, R., Ney, H. 1995. Improved backing-off for m-gram language modeling. In: Proceeding of International Conference on Acoustics, Speech, and, Signal Processing, pp. 181–184.

- Koerich, A.L., 2004. Rejection strategies for handwritten word recognition. In: Proceedings of 9th International Workshop on Frontiers in Handwriting Recognition, pp. 479–484.
- Koerich, A.L., Sabourin, R., Suen, C., 2003. Large vocabulary off-line handwriting recognition: a survey. *Pattern Anal. Appl.* 6, 97–121.
- Lam, L., Lee, S.W., Suen, C.Y., 1992. Thinning methodologies—a comprehensive survey. *IEEE Trans. PAMI* 14, 869–885.
- Likforman-Sulem, L., Zahour, A., Taconet, B., 2007. Text line segmentation of historical documents: a survey. *Int. J. Doc. Anal. Recogn.* 9, 123–138.
- Lorette, G., 1999. Handwriting recognition or reading? What is the situation at the dawn of the 3rd millennium? *Int. J. Doc. Anal. Recogn.* 2, 2–12.
- Marti, U.V., Bunke, H., 2001. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *Int. J. Pattern Recogn. Artif. Intell.* 15, 65–90.
- Marti, U.V., Bunke, H., 2002. The IAM-database: an English sentence database for off-line handwriting recognition. *Int. J. Doc. Anal. Recogn.* 5, 39–46.
- Marukatat, S., Artières, T., Gallinari, P., Dorizzi, B., 2002. Rejection measures for handwriting sentence recognition. In: Proceedings of 8th International Workshop on Frontiers in Handwriting Recognition, pp. 24–29.
- Neyman, J., Pearson, E.S., 1933. On the problem of the most efficient tests of statistical hypotheses. *Philos. Trans. Roy. Soc. Lond. Ser. A* 231, 289–337.
- Pitrelli, J., Subrahmonia, J., Perrone, M.P., 2006. Confidence modeling for handwriting recognition: algorithms and applications. *Int. J. Doc. Anal. Recogn.* 8, 35–46.
- Plamondon, R., Privitera, C.M., 1999. The segmentation of cursive handwriting: an approach based on off-line recovery of the motor-temporal information. *IEEE Trans. Image Process.* 8, 80–91.
- Plamondon, R., Srihari, S., 2000. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Trans. PAMI* 22, 63–84.
- Ploetz, T., Fink, G.A., 2009. Markov models for offline handwriting recognition: a survey. *Int. J. Doc. Anal. Recogn.* 12, 269–298.
- Pratikakis, I., Gatos, B., Ntirogiannis, K., 2010. H-DIBCO 2010 – handwritten document image binarization competition. In: Proceedings of 12th International Conference on Frontiers in Handwriting Recognition, pp. 727–732.
- Rabiner, L., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 257–285.
- Rodriguez, J., Perronnin, F., 2008. Local gradient histogram features for word spotting in unconstrained handwritten documents. In: Proceeding of 1st International Conference on Frontiers in Handwriting Recognition, pp. 7–12.
- Rodriguez, J., Perronnin, F., 2009. Handwritten word-spotting using hidden Markov models and universal vocabularies. *Pattern Recogn.* 42, 2106–2116.
- Rose, R., Juang, B., Lee, C., 1995. A training procedure for verifying string hypotheses in continuous speech recognition. In: Proceeding of International Conference on Acoustics, Speech, and, Signal Processing, pp. 281–284.
- Saleem, S., Cao, H., Subramanian, K., Kamali, M., Prasad, R., Natarajan, P., 2009. Improvements in BBN's HMM-based offline arabic handwriting recognition system. In: Proceeding of International Conference on Document Analysis and Recognition, pp. 773–777.
- Saon, G., 1999. Cursive word recognition using a random field based hidden Markov model. *Int. J. Doc. Anal. Recogn.* 1, 199–208.
- Sayre, K.M., 1973. Machine recognition of handwritten words: a project report. *Pattern Recogn.* 5, 213–228.
- Schambach, M., 2008. Fast script word recognition with very large vocabulary. In: Proceeding of International Conference on Document Analysis and Recognition, 2008, pp. 9–13.
- Schlappbach, A., Bunke, H., 2004. Off-line handwriting identification using HMM-based recognizers. In: Proceedings of 17th International Conference on Pattern Recognition, pp. 654–658.
- Schlappbach, A., Bunke, H., 2006. Off-line writer verification: A comparison of a hidden Markov model (HMM) and a Gaussian mixture model (GMM) based system. In: Proceedings of 10th International Workshop on Frontiers in Handwriting Recognition, pp. 275–280.
- Srihari, S.N., 2000. Handwritten address interpretation: a task of many pattern recognition problems. *Int. J. Pattern Recogn. Artif. Intell.* 14, 663–674.

- Srihari, S., Shin, Y., Ramanaprasad, V., 1996. A system to read names and addresses on tax forms. Proc. IEEE 84, 1038–1049.
- Su, B., Lu, S., Tan, C.L. 2010. Binarization of historical document images using the local maximum and minimum. In: Proceedings of 9th International Workshop on Document Analysis Systems, pp. 159–166.
- Tarazón, L., Pérez, D., Serrano, N., Alabau, V., Ramos Terrades, O., Sanchis, A., Juan, A., 2009. Confidence measures for error correction in interactive transcription handwritten text. In: Proceedings of 15th International Conference on Image Analysis and Processing, pp. 567–574.
- Thomas, S.C., Heutte, L., Paquet, T. 2010. An information extraction model for unconstrained handwritten documents. In: Proceedings of 20th International Conference on Pattern Recognition, pp. 3412–3415.
- Tortorella, F. 2000. An optimal reject rule for binary classifiers. In: Advances in Pattern Recognition. Springer, LNCS, vol. 1876, pp. 611–620.
- Toselli, A.H., Romero, V., Pastor, M., Vidal, E., 2010. Multimodal interactive transcription of text images. Pattern Recogn. 43, 1814–1825.
- Trier, O.D., Taxt, T., 1995. Evaluation of binarization methods for document images. IEEE Trans. PAMI 17, 312–315.
- Varga, T., Bunke, H., 2004. Off-line handwritten word recognition using synthetic training data produced by means of a geometrical distortion model. Int. J. Pattern Recogn. Artif. Intell. 18, 1285–1302.
- Vinciarelli, A., Bengio, S., 2002. Off-line cursive word recognition using continuous density HMMs trained with PCA and ICA features. In: 16th International Conference on Pattern Recognition, pp. 81–84.
- Vinciarelli, A., Luettin, J., 2000. Off-line cursive script recognition based on continuous density HMM. In: Proceedings of 7th International Workshop on Frontiers in Handwriting Recognition, pp. 493–498.
- Vinciarelli, A., Luettin, J., 2001. A new normalization technique for cursive handwritten words. Pattern Recogn. Lett. 22, 1043–1050.
- Vinciarelli, A., Bengio, S., Bunke, H., 2004. Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. IEEE Trans. PAMI 26, 709–720.
- Wagner, R., Fischer, M., 1974. The string-to-string correction problem. J. Assoc. Comput. Mach. 21, 168–173.
- Wessel, F., Schlüter, R., Macherey, K., Ney, H. 2001. Confidence measures for large vocabulary continuous speech recognition. IEEE Trans. Speech Audio Process. 9, 288–298.
- Wienecke, M., Fink, G., Sagerer, G., 2005. Toward automatic video-based whiteboard reading. Int. J. Doc. Anal. Recogn. 7, 188–200.
- Zimmermann, M., Bunke, H. 2002. Hidden Markov model length optimization for handwriting recognition systems. In: Proceedings of 8th International Workshop on Frontiers in Handwriting Recognition, pp. 369–374.
- Zimmermann, M., Bunke, H. 2004. N-gram language models for offline handwritten text recognition. In: Proceedings of 9th International Workshop on Frontiers in Handwriting Recognition, pp. 203–208.
- Zimmermann, M., Chappelier, J.C., Bunke, H., 2006. Offline grammar-based recognition of handwritten sentences. IEEE Trans. PAMI 28, 818–821.

Machine Learning in Handwritten Arabic Text Recognition

Utkarsh Porwal, Zhixin Shi, and Srirangaraj Setlur

University at Buffalo, The State University of New York, USA

Abstract

Automated recognition of handwritten text is one of the most interesting applications of machine learning. This chapter poses handwritten Arabic text recognition as a learning problem and provides an overview of the ML techniques that have been used to address this challenging task. The use of co-training for solving the problem of paucity of labeled training data and structural learning approaches to capture contextual information for feature enhancement have been presented. A system for recognition of Arabic PAWs using an ensemble of biased learners in a hierarchical framework and the use of techniques such as Artificial Neural Networks, Deep Belief Networks, and Hidden Markov models within this hierarchical framework to improve text recognition have been described. The chapter also describes some of the features that have been successfully used for handwritten Arabic text classification for completeness since the selection of discriminative features is critical to the success of the classification task.

Keywords: OCR, handwriting recognition, arabic script, learning algorithms, features for text recognition, cotraining, structural learning, HMM, ensemble of learners, neural networks, deep belief networks

1. Introduction

The field of automated handwriting recognition has achieved significant real world success in targeted applications such as address recognition on mail-pieces for sorting automation and reading of courtesy and legal amounts on bank checks, by using domain-dependent constraints to make the problem tractable. However, unconstrained handwritten text recognition is still an open problem which is attracting renewed interest as an active area of research due to the proliferation

of smartphones and tablet devices where handwriting with finger or stylus is likely to be a potentially convenient mode of input for these handheld devices. Much of the past research on handwritten text recognition has been focused on Latin and CJK (Chinese, Japanese, Korean) scripts while scripts such as Arabic and Devanagari are now seeing increased interest. In this chapter, we discuss machine learning approaches in the context of handwritten Arabic text recognition.

OCR of handwritten Arabic script, in particular, has proven to be a very challenging problem. This chapter poses Arabic text recognition as a learning problem and investigates the fundamental challenges for learning algorithms. Techniques to overcome such challenges are covered in detail along with the state-of-the-art methods proposed for handwritten Arabic text recognition. Handwritten text recognition is a hard problem due to several additional challenges such as variations within the writing of a single writer as well as between writers and noise in the data. Systems developed for recognition of Latin script have not been entirely successful in the recognition of Arabic text due to script-specific challenges such as the highly cursive nature of Arabic script, heavy use of dots and other diacritic marks, and context dependent variations in the shape of some characters. This is partly due to the fact that features that are discriminative for Latin text are not necessarily discriminative for Arabic text. While the main focus of the chapter is learning algorithms, we also briefly describe features that have worked well for Arabic text recognition (Section 4). Language models have also been successfully used to augment recognition performance but have not been addressed in this chapter.

2. Arabic script—challenges for recognition

The Arabic language is widely spoken in many parts of the world and the Arabic script is used for representing Arabic and related languages such as Persian, Urdu, Pashto, Sindhi, and Kurdish, as well as many African languages. The Arabic script is cursive, even in its printed form, and hence the same character can be written in different forms depending on how it connects to its neighbors (*ligatures*). The Arabic script is written from right to left and the alphabet consists of 28 letters each of which has between two to four context or position dependent shapes within a word or sub word. The shapes correspond to the beginning, middle, or end of the (sub-) word or in isolation. Figure 1 shows the list of all Arabic characters with the corresponding forms that they can take based on their position within the text.

However, a few letters do not have initial or medial forms and cannot be connected to the following letters. When these letters occur in the middle of the word, the word is divided into sub words often known as parts of Arabic words or PAWs. Figure 2 shows Arabic words with a combination of sub words. Each sub word is usually a single connected component with small unconnected diacritical marks.

Additionally, other diacritics are used to represent short vowels, syllable endings, nunciation, and doubled consonants (Lorigo and Govindaraju, 2006). Such diacritics (hamza and shadda) are shown circled in Fig. 3.

Arabic script exhibits a strong baseline along which it is written from right to left. Some letters have *ascenders* and *descenders*. Usually characters are joined along this

Name	Isolated	Initial	Medial	Final
alif	ا	-	-	ا
baa	ب	ب	ب	ب
taa	ت	ت	ت	ت
thaan	ث	ث	ث	ث
jiim	ج	ج	ج	ج
Haa	ح	ح	ح	ح
khaa	خ	خ	خ	خ
daal	د	-	-	د
dhaal	ذ	-	-	ذ
raa	ر	-	-	ر
zaay	ز	-	-	ز
siin	س	-	-	س
shuin	ش	ش	ش	ش
Saad	ص	ص	ص	ص
Daad	ض	ض	ض	ض
Taa	ط	ط	ط	ط
Dhaa	ظ	ظ	ظ	ظ
ayn	ع	ع	ع	ع
ghayn	غ	غ	غ	غ
faa	ف	ف	ف	ف
qaaf	ق	ق	ق	ق
kaaf	ك	ك	ك	ك
laam	ل	ل	ل	ل
muim	م	م	م	م
nuun	ن	ن	ن	ن
haa	ه	ه	ه	ه
waaw	و	-	-	و
yaa	ي	ي	ي	ي

Fig. 1. Arabic alphabet.



Fig. 2. Arabic words with one, two, three, and four sub words.



Fig. 3. Handwritten words with diacritics—Hamza and Shadda.

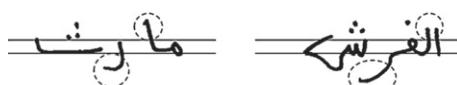


Fig. 4. Handwritten Arabic words with ascenders and descenders.

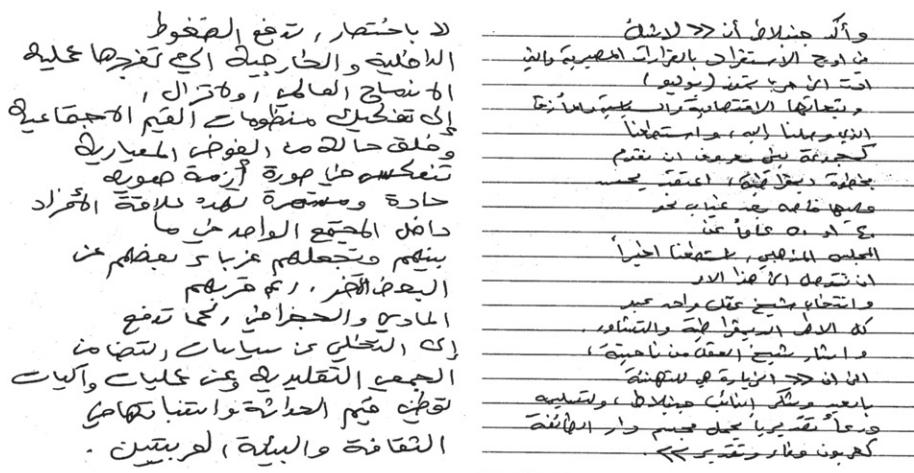


Fig. 5. Sample handwritten Arabic documents.

baseline. Figure 4 shows Arabic words with *ascenders* and *descenders* along with the baseline.

Many of these characteristics of Arabic text make recognition a difficult task. Long strokes parallel to the baseline and the ligatures that also run along the baseline make it difficult to segment Arabic words into their component characters. As a result, techniques developed for Latin scripts often do not translate well to Arabic script. Figure 5 shows two sample handwritten Arabic documents that illustrate the characteristic features of Arabic script. Over the past two decades many approaches have been tried to recognize Arabic text. Simple rule based methods such as template matching do not work well with handwritten documents given the variability even within a single writer's handwriting. Writers have different writing styles and a few templates cannot capture all variations. In the last decade, the focus has been on trying to develop principled machine learning techniques to solve the problem of handwriting recognition. A typical approach is to formulate the text recognition task as a supervised learning problem in which a classifier is trained to distinguish list of classes. Data samples that are labeled (their classes are known in advance) are provided to the learning algorithm to train a classifier and the trained classifier will then be used to recognize the class or label of the test data sample. However, performance of the classifier will depend on several factors with the key ones being (1) amount of training data provided, (2) similarity of the test data to the training data, (3) inductive bias of the classifier (assumptions made by classifier to discriminate between member classes), and (4) the quality and amount of information provided to the learning algorithm for learning (feature extraction and availability). Out of these four factors, (2) is more of a requirement than constraint. No model can succeed if it is tested on a data that is very different from the data it was trained on. Therefore, it is required in machine learning problems that training and test data should come from the same data distribution. However, some relaxation of these requirements are addressed in transfer learning (Pan and Yang, 2010).

One of the primary challenges in Arabic handwriting recognition is obtaining labeled data because the process of annotating the data is tedious and expensive. Annotating data requires human intervention which makes the whole process of labeling very slow. It is often difficult to collect sufficient data pertaining to each class (word or character) so that a classifier can be learned. Obtaining labeled data is a bottleneck for handwriting recognition. However, unlabeled data is easy to obtain as documents can be scanned at minimal cost. Therefore, it is prudent to investigate if unlabeled data can be used to improve the performance of the learning algorithm. This setting where unlabeled data is used along with labeled data for learning is referred to as semi-supervised learning in the literature and some learning paradigms that are relevant to the recognition task are introduced in Section 3.

Another challenge is of feature selection and good discriminative features are crucial for optimal performance of the learning algorithm. However, it is often difficult to capture all the information present in the data in terms of features as some of the information is contextual or domain specific. Therefore, it is a challenge to capture such information and can undermine the performance of the learning algorithm if all the information present in the training data cannot be leveraged. Some of the features that have been effective in the recognition of handwritten Arabic documents are described in Section 4.

A third important factor is the selection of the model as it plays an important role in the performance of recognition or prediction system. Some learners are better suited for certain types of data although it does not necessarily guarantee better performance. For instance, learners like HMM (Rabiner, 1990) and DTW (Puurula and Compernolle, 2010) are a natural fit for capturing temporal information (1D data) such as in the case of speech recognition because of their assumptions and formulations. Likewise, techniques such as MRF (Li, 1995) and CRF (Wang and Ji, 2005) are ideally suited to capture spatial information (2D data) in images or videos. However, sometimes no single classifier may be good enough for the task and the selection of algorithms becomes non-trivial as it is difficult to make any a priori assumptions about the structure of the data. In such instances, the selection of a model that will work with all constraints such as limited labeled data or missing information becomes a challenge. In this chapter, we seek to investigate techniques to address all of these issues and specific models that have worked well in the domain of handwritten text recognition are described in Section 5.

3. Learning paradigms

A learning problem can be formally defined as finding the mapping function between the input vector $\mathbf{x} \in \mathbf{X}$ and its output labels $y \in \mathbf{Y}$. In the training phase, a finite number of data points (\mathbf{x}_i, y_i) are provided under the assumption that they are all drawn from some unknown probability distribution Ω in domain \mathbf{D} . The function that takes an input vector \mathbf{x} and produces the output label y is called the *target concept*. In the process of locating this target concept, a learner will output the hypothesis which is consistent with most of the training samples while minimizing the error. Therefore, formally a learner will output

$$h^* = \underset{h \in \mathbf{H}}{\operatorname{argmin}} \{\operatorname{err}(h)\}. \quad (1)$$

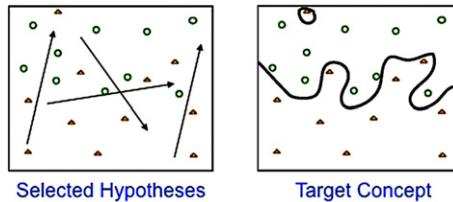


Fig. 6. Selected hypothesis and actual target concept.

Any learner will explore the hypotheses space and output a hypothesis, therefore selection of the most suitable learning algorithm is crucial in this regard. Consider the case shown in Fig. 6 where the learning algorithm selected is linear and the actual target concept is some nonlinear function. In this scenario, the learning algorithm can never locate the target concept regardless of the number of training samples provided to explore the hypotheses space because the generated hypothesis is linear and target concept is nonlinear. Therefore, selection of the right hypotheses space (learning algorithm) is very important for any learner to succeed. Domain knowledge and context-based assumptions can help in the selection of the right model and optimizing the performance of the algorithm. Error due to bad quality of hypotheses space is called *approximation error*.

It is also possible that no target concept exists that can correctly generate labels for all the training data samples. This might be due to inherent noise in the data. One data point can also have multiple labels due to noise, therefore no function will be able to generate those labels. Even assuming that a target concept does exist, it can still be difficult to locate it due to several reasons. The learner explores the hypotheses space by minimizing the error over number of data samples. Therefore, more data samples should help in better exploration of the hypotheses space. However, it is often difficult to get enough data samples in practice, as target concepts can be very high dimensional complex functions and vast amounts of training data will be needed to explore the hypotheses space. Error due to limited number of samples is called *estimation error*. Therefore, limited number of data samples is the second challenge after selection of the right model.

A third issue of concern is the nonavailability of sufficient information. Here sufficient may refer to not only the quantity of information but also the right information. Imagine the actual target concept is a m dimensional function and the feature extracted for classification is n dimensional. If $m > n$ then in an attempt to locate the target function, the learner will wander in a n dimensional space while the actual function lies in a m dimensional space. Therefore, the best any learner can do is to approximate the target concept in n dimensional space; i.e. the projection of the actual function in a lower dimensional space. However, if more features (information) can be provided, then it is possible to search for the target function in either the ideal feature space or in a space where loss of information is minimal. It is also possible that $m = n$ or $m < n$. Even if the dimensionality of the target function and the extracted feature are the same, it is possible that the target concept is in a different space. The additional information in this case will help in getting close to

the space of target concept (for example, by rotation in the presence of skew). A similar argument holds for $m < n$, where it is possible that the target concept is of a lower dimension but the feature is not capturing enough information about the dimensions of the target concept.

Although learning theory has many different schools of thought, one of the simplest yet powerful paradigms is Probably Approximately Correct (PAC) learning (Valiant, 1984). In PAC setting, any learner h from \mathbf{H} is a PAC learner if for *any* ϵ where $0 < \epsilon < 1/2$, for *any* δ where $0 < \delta < 1/2$ and for *any* distribution Ω in domain \mathbf{D}

$$\text{Prob}[\text{err}_\Omega(h) \leq \epsilon] \geq 1 - \delta, \quad (2)$$

where the error is calculated over number of data points $\{\mathbf{x}_i, c(\mathbf{x}_i)\}$ for $i \in [1, N]$ sampled from the distribution Ω as shown below

$$\text{err}_\Omega(h) := \text{Prob}_{(\mathbf{x}) \sim \Omega}[h(\mathbf{x}) \neq c(\mathbf{x})]. \quad (3)$$

However, PAC model of learning has very strong assumptions and it is likely that in real-world applications such assumptions will not hold true. It assumes that a function that can generate labels for all the data points exists for *any* distribution in domain \mathbf{D} and for *any* $0 < \epsilon, \delta < 1/2$. This assumption is very strict as this may not be true for *every* distribution, ϵ or δ . Moreover, it might be possible that such a target concept may not exist which generates all the labels because of the noise in the data and even if there is such function, locating it might be a NP hard problem. To circumvent these limitations, the learning paradigm known as Inconsistent Hypothesis Model (IHM) was proposed. In this model, training data points $\{\mathbf{x}_i, y_i\}$ for $i \in [1, N]$ are sampled from *some* distribution Ω over $\mathbf{D} \times \{0, 1\}$ (considering only a two-class case) where $\mathbf{x}_i \in \Omega$ and output labels $y_i \in \{0, 1\}$. The learning algorithm will select a hypothesis h from \mathbf{H} such that this hypothesis minimizes the *generalization error*

$$\text{err}_\Omega(h) := \text{Prob}_{(\mathbf{x}, y) \sim \Omega}[h(\mathbf{x}) \neq y]. \quad (4)$$

Any learner will output the most optimal hypothesis:

$$h^* = \underset{h \in \mathbf{H}}{\operatorname{argmin}} \{\text{err}(h)\}. \quad (5)$$

Having formally described the learning algorithms, the next subsection will address the challenge of limited data samples by using co-training, a PAC style semi-supervised algorithm.

3.1. Co-training

Blum and Mitchell (1998) proposed co-training as a method which needs small amount of labeled training data to start. It requires two separate views of the data $\mathbf{X} = \mathbf{X}_1 \times \mathbf{X}_2$ and both of the views should be orthogonal to each other i.e., each view should be sufficient for correct classification. **Blum and Mitchell (1998)** gave a PAC style analysis of this algorithm where data samples $x = \{x_1, x_2\}$ are drawn from some distribution \mathbf{D} and there are two target concepts $f_1 \in \mathbf{C}_1$ and $f_2 \in \mathbf{C}_2$

corresponding to each view such that $f_1(x_1) = f_2(x_2)$. Therefore, if f is considered as the combined target concept for any data point x drawn from distribution \mathbf{D} with nonzero probability then $f(x) = f_1(x_1) = f_2(x_2) = y$ (label). Here, if $f_1(x_1) \neq f_2(x_2)$ for some x with nonzero probability assigned by \mathbf{D} then (f_1, f_2) are considered incompatible with distribution \mathbf{D} . Therefore, co-training will use unlabeled data to find the best compatible pair (h_1^*, h_2^*) with the distribution \mathbf{D} .

Therefore, two learners will be trained on the labeled data available initially for the two views and they will reiteratively label some unlabeled data points. In each round of co-training, a cache is drawn from the unlabeled data set and all the data points in the cache are labeled. The learner labels all the data points with a certain degree of confidence and some of the data points are selected from this cache and added to the training set. Selection of these data points is crucial since the performance of the learner will increase only if the added points have correct labels. Newly added points should be such that they increase the confidence of the learner in making decisions about labels of data points in the next iteration. If incorrectly labeled data points are added, the performance of the learner will likely decrease. Given the snowballing effect in each iteration, the performance of the selection algorithm will decide the robustness of the algorithm. Therefore, co-training is an iterative bootstrapping method which seeks to increase the confidence of the learner in each round. It boosts the confidence of the score much as the Expectation Maximization (EM) method does but works better than EM ([Nigam and Ghani, 2000b](#)). In EM, all the data points are labeled in each round and the parameters of the model are retuned. This is done till convergence is achieved, i.e., when parameters do not change with new information whereas in co-training a few of the data points are labeled each round and the classifiers are then retrained. This helps build a better learner in each iteration which in turn would lead to better decisions and hence an increase in the overall accuracy of system.

[Blum and Mitchell \(1998\)](#) showed that co-training works best if the two views are orthogonal to each other and each of them is capable of classification independently. They showed that if the two views are conditionally independent and each learner trained on the views is a low error classifier then the accuracy of classifiers can be increased significantly. They proved that error rates of both the classifiers decreases during co-training because of the extra information added to the system in each round. This extra information directly depends on the degree of un-correlation. This is because the system is using more information to classify data points. Since both views are independently sufficient for classification, this brings redundancy while producing more information. However, [Abney \(2002\)](#) later reformulated the metric for co-training in terms of the measure of agreement between learners over unlabeled data. [Abney \(2002\)](#) gave an upper bound on the error rates of learners based on the measure of their disagreement. However, [Nigam and Ghani \(2000a\)](#) proved that completely independent views are not required for co-training and that it works well even if the two views are not completely uncorrelated.

3.1.1. Selection algorithm

Several selection algorithms have been proposed based on the kind of data and application. One approach is to select a set of data points randomly and add to the

labeled set (Clark et al., 2003). The system is retrained and its performance is tested on the unlabeled data. This process is repeated for some iterations and performance on each set of points is recorded. The set of points resulting in the best performance are selected to be added to the labeled set and the rest are discarded. This method is based on the degree of agreement of both learners over unlabeled data in each round. Other methods that have been tried include choosing the top k elements from the newly labeled cache, selecting the ones with maximum scores, or choosing some maximum and some minimum scoring points (Wang et al., 2007). Other heuristics include checking the standard deviation or just choosing a fixed number of top points in every round. In all these cases, an empirically determined threshold is used for the selection criteria and since this is not a very principled approach, the efficacy of the method is dependent on the kind of data or application. Porwal et al. (2012) proposed an oracle-based selection approach where selection was made without using any heuristics. The approach was based on learning the pattern of the distribution of scores for different classes or writers given by the learners. If this pattern can be learned, then for any unseen data point, the oracle would be able to predict the class or label from the score distribution generated by the learner. The advantage of this approach would be a robust selection algorithm that would work regardless of any specific data or application.

3.1.2. Oracle training

A validation set is used for the training of the oracle and the trained oracle is used for selecting data points after each round. Training of the oracle is done before co-training starts. A classifier is trained on the initial training set and its performance is tested on the validation set. Now, the score distribution over all classes is considered as feature and an oracle classifier is trained using these features. Data points for which the predicted class matches the ground truth are assigned to the positive class and the rest of the data points is assigned to the negative class. The new features are score distributions over all classes and all data points are divided into two new classes, viz. positive and negative. The task is now narrowed down to a two-class problem where one class has all the points that meet the selection criteria and the other class has the data points which should be discarded. Here, the oracle classifiers need not be same as the learner used in co-training process.

Once the oracle is trained, co-training begins. In each round of co-training, a cache is selected from the unlabeled dataset and is tested against the learner. Some of the data points with best performance are added to the trained set and the learners are retrained. These data points from the cache will be selected by the oracle. After each round, a score distribution of all data points will be generated. These scores are considered as features of a new test set for the oracle. This set would be tested against the oracle and it would label all the data points with two new classes that are either positive or negative. All the data points classified as positive by the oracle are selected for addition to the training set. Learners are retrained and the second round of co-training resumes and this process repeats until the unlabeled dataset is exhausted. Here the upper limit on the performance of the oracle depends on the accuracy of the learner. If the oracle selects all the data points correctly labeled by the learner, the accuracy of performance is still dependent on the performance of the

learner. Hence, the upper bound on the performance is determined by the accuracy of the learner.

Co-training is useful for generating labeled samples for handwritten text recognition and addresses the first fundamental problem of learning algorithms i.e., availability of limited number of data samples. Using co-training, one can generate labels for unlabeled data and thus aid in enhancing the accuracy of the recognition system. However, even when sufficient labeled data is available, the task of Arabic handwritten text recognition is not trivial because of the variation present in the data. Handwriting of multiple writers can be clustered into some broad writing styles. Handwriting of an individual captures several nuances of a writer's personality and background (Shivram et al., 2012). There are factors influencing the handwriting of an individual which are abstract, such as the effect of the native language on the writing of the nonnative languages also known as accent of an individual (Ramaiah et al., 2012). While features that capture these abstract notions are critical for applications such as writer identification, the challenge in handwritten text recognition is to find features that can tolerate the wide variation in handwriting while being reliably discriminative between the classes. Porwal et al. (2012) proposed a novel approach for feature enhancement in semi supervised framework using structural learning to capture the information that was difficult to extract through regular feature extraction techniques.

3.2. Structural learning

Porwal et al. (2012) proposed a structural learning-based approach for feature enhancement where a target task is divided into several related auxiliary tasks (Ando and Zhang, 2005; Blitzer et al., 2006). These auxiliary tasks are then solved and a common structure is retrieved, which in turn is used to solve the original target task. This approach, also known as multi-task learning in the machine learning literature, is very effective in a semi-supervised framework where labeled data is limited and the original problem is complex and rich enough to be broken down into sub problems and each candidate sub problem provides information useful for solving the target problem. As discussed above, any learner will explore the hypotheses space to approximate the target function using training data samples. Since usually, limited data points are available to explore any hypotheses space, selection of appropriate and rich space is central to the performance of the learner. Often learner fails to approximate the target function because it does not lie within the space that the learning algorithm is exploring. The central idea of structural learning is to select the most appropriate hypotheses space with the use of finite labeled data available.

The key concept of structural learning is to break down the main task into several related tasks and then find a common low dimensional optimal structure which has high correspondence with every sub task. This structure is used to solve the main problem. The optimal structure would correspond to the scenario where the cumulative error of all the sub tasks will be minimized. This optimal structure captures information that is domain specific and is very useful in solving the main task as it helps in the selection of the right hypotheses space. e.g., the nuances of

handwritten text captured by accent, styles, etc. can be considered as related sub tasks of the main task of handwritten text recognition. Since several such aspects of handwriting are abstract in nature, there needs to be a principled way to define these sub tasks. [Ando and Zhang \(2005\)](#) propose an approach to create such related sub tasks in a semi-supervised framework.

The efficacy of structural learning lies in the fact that in almost all of the real-world problems the hypothesis produced by an algorithm is a smooth discriminant function. This function maps points in the data domain to the labels. The smoothness of this function is enforced by a good hypotheses space. If any two points are close in the domain space then the mapping produced by the discriminant function will also be close in the target space. Therefore, if one can find such discriminant functions then this implies a good hypotheses space.

In structural learning, we find several such functions that correspond to the structure of the underlying hypotheses space. If the sub tasks are related we get information about context embedded in the optimal structure. If sub tasks are not related, the structural parameter still contains information about the smoothness of the hypotheses space. Therefore, breaking the main task into sub tasks is helpful even though they are not related as the structure retrieved will still have the information about smoothness of the space.

Formally, structural learning can be defined as a collection of T sub tasks indexed by $t \in \{1, \dots, T\}$ and each sub task has n_t samples over some unknown distribution Ω_t . All the sub tasks have their respective candidate hypotheses spaces $\mathbf{H}_{\theta,t}$ indexed by the parameter θ which is common to all the sub tasks and encapsulate all the information that are useful for solving the primary task. The new objective function is to minimize the joint empirical error

$$h_{\theta,t}^* = \operatorname{argmin}_{h \in \mathbf{H}_{\theta,t}} \sum_{i=1}^{n_t} L(h(\mathbf{x}_i^t), y_i^t), \quad (6)$$

where L is the loss function.

The first step is to create the auxiliary tasks related to the main task. Auxiliary tasks can be formulated as capturing abstract aspects of the main tasks which cannot be well formulated but are still vital in recognition of the handwritten text. One way to create auxiliary tasks in a semi-supervised framework is by making use of unlabeled data. However, creation of an auxiliary task should address two issues. First is the label generation for the auxiliary tasks. The process should generate automatic labels for each auxiliary task. Second condition is of relevancy among the auxiliary tasks. It is desirable that the auxiliary tasks are related to each other so that a common optimal structure can be retrieved. [Ando and Zhang \(2005\)](#) suggested few generic methods to create auxiliary tasks that would satisfy these two conditions. We will cover one of those techniques in this chapter.

In this method two distinct features ϕ_1 and ϕ_2 are used. First, a classifier is trained for the main task using the feature ϕ_1 over labeled data. Same feature is extracted from unlabeled data and the classifier trained is used to create auxiliary labels for the unlabeled data. The auxiliary task is to create binary classification problems

Algorithm 1 Structural Learning Algorithm

Require:

- 1: $X1 = [\phi_1(\mathbf{x})_t, y_t]_{t=1}^T \leftarrow$ Labeled Feature One
- 2: $X2 = [\phi_2(\mathbf{x})_t, y_t]_{t=1}^T \leftarrow$ Labeled Feature Two
- 3: $U = [\phi_2(\mathbf{x})_j] \leftarrow$ Unlabeled Data Feature Two
- 4: $C \leftarrow$ Classifier
- 5: Train C with $X1$
- 6: Generate auxiliary labels by labeling U with C
- 7: For a L class problem create L binary prediction problems as auxiliary tasks, $y_l = h_l \phi_2(\mathbf{x}), l = 1 \dots L$
- 8: **for** $l = 1 \dots L$ **do**
- 9: $\mathbf{w}_{l,\theta} = (\phi_2(\mathbf{x})^T \phi_2(\mathbf{x}))^{-1} \phi_2(\mathbf{x})^T y_l$
- 10: **end for**
- 11: $\mathbf{W} = [\mathbf{w}_1 | \dots | \mathbf{w}_L]$
- 12: $[\mathbf{U} \Sigma \mathbf{V}^T] = SVD(\mathbf{W})$
- 13: Projection onto \mathbb{R}^h , $\Theta = \mathbf{U}_{[:,1:h]} = [\theta_1 | \dots | \theta_h]$
- 14: New feature in \mathbb{R}^{N+h} space, $[\phi_2(\mathbf{x}) \theta \phi_2(\mathbf{x})]$

for predicting the label assigned for each of the data points in the unlabeled data. Therefore, for a n class problem as the main task n auxiliary tasks can be created as a two-class problem. An auxiliary predictor will give label 1 if it can predict the correct auxiliary label, otherwise it will assign 0. Any auxiliary predictor can be written as

$$h_{\mathbf{w}}(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \dots + w_n x_n \quad (7)$$

and the goal is to reduce the empirical error as given by Eqs. (5) and (6). Therefore, error can be written as

$$y = h(\mathbf{w}, \mathbf{x}) + \epsilon. \quad (8)$$

In order to minimize this error, we take least square loss function and minimize the joint empirical error for all the data points

$$\text{err}_{\Omega}(h) = \frac{1}{2} \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x})^2. \quad (9)$$

To minimize the joint empirical error, the algorithm seeks the optimal weight vector. Setting gradient of the error function to zero will give the optimal weight vector

$$0 = \sum_{i=1}^n y_i \mathbf{x}^T - \mathbf{w}^T \left(\sum_{i=1}^n \mathbf{x} \mathbf{x}^T \right). \quad (10)$$

Solving for \mathbf{w} we obtain

$$\mathbf{w}_{\text{opt}} = (\mathbf{x}^T \mathbf{x})^{-1} \mathbf{x}^T y. \quad (11)$$

This will give optimal weights for one predictor of one auxiliary task. To get the optimal structure corresponding to all the sub tasks this process should be repeated for all the auxiliary tasks. After the optimal \mathbf{x} is calculated for all the auxiliary tasks, a big weight matrix \mathbf{W} of all such weight vectors is created whose columns are the weight vectors of the hypothesis of auxiliary classes.

Once the big weight matrix \mathbf{W} is calculated, it can be used to find the low dimensional common sub space. However, before doing dimensionality reduction, redundancy in the information is removed. Often sub tasks are related to each other along with the main task and they capture information of the same nature. Thus, it may not add much to discriminatory power of the algorithm to solve the main task. Since, information hidden in the weight vectors could be related, only left singular vectors are picked from the singular value decomposition (SVD) of the \mathbf{W} matrix. Therefore,

$$[\mathbf{U} \Sigma \mathbf{V}^T] = SVD(\mathbf{W}). \quad (12)$$

Initially, weight vectors are in the feature space \mathbb{R}^N but they can be projected onto some lower dimensional space \mathbb{R}^h to capture the variance of the auxiliary hypotheses space in the best h dimension. Therefore the low dimensional feature mapping is $\theta^T \mathbf{x}$. This new feature mapping can be appended to the original feature vector to solve the main task in \mathbb{R}^{N+h} space. Thus, structural learning can be used to capture the contextual information which is difficult to extract otherwise by regular feature extraction techniques.

The next section describes some of the features that have been successfully used for handwritten Arabic text recognition.

4. Features for text recognition

Determination of an appropriate set of features that can capture the discriminative characteristics of Arabic characters/words while allowing for variability in handwriting is a key challenge for handwritten Arabic OCR. Feature extraction typically involves the conversion of a two-dimensional image into a one-dimensional feature sequence or feature vector. Selection of good discriminative features is critical to achieve high recognition performance. Survey on evaluations of features for hand printed isolated characters can be found in Trier et al. (1995) and Liu et al. (2003). In this section, we describe a few features that have proven to be very effective in the recognition of handwritten cursive Arabic text.

4.1. Gradient-structural-concavity (GSC) features

The design philosophy underlying these features is to capture the characteristics of handwritten character images at multiple resolutions, from fine to coarse. The GSC features capture the local, intermediate, and global characteristics of a handwritten image (Favata et al., 1994). Specifically, the gradient captures the local stroke shape, structural features capture the coarser trajectory of the stroke and the concavity features encapsulates stroke relationships at an even coarser level (see Fig. 7). While the original features were binary in nature, re-implementation using floating point numbers provides better performance.

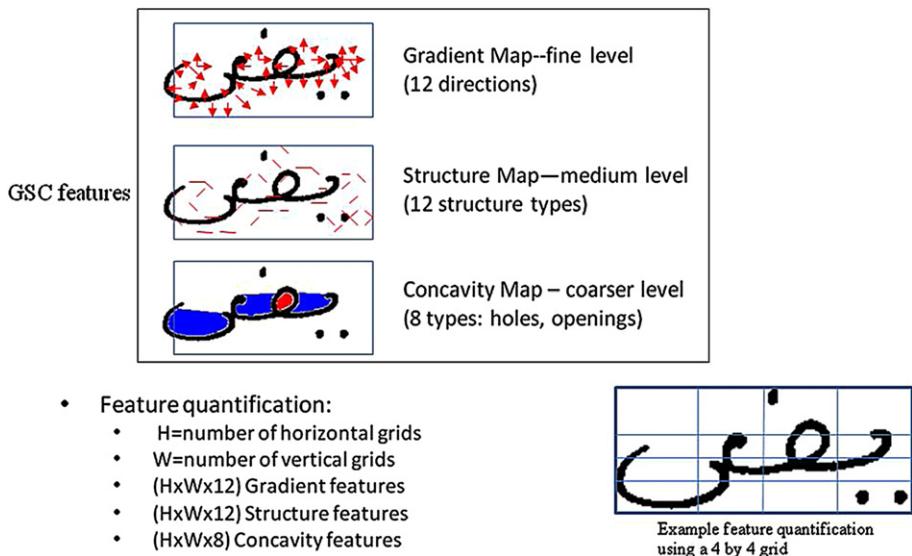


Fig. 7. GSC features.

The gradient features are computed by applying a Sobel operator on the binary character/word images. The operator generates an approximation of the x and y derivatives at each image pixel. By definition, the gradient is a vector with the direction ranging from 0° to 359° . The range is split into 12 non-overlapping regions and sampled into bins (for example a $m \times n$ grid). A histogram is computed in each gradient direction at each pixel within a bin. When a $m \times n$ grid is used, a vector of $12 \times m \times n$ floating point numbers represents the gradient feature of the image.

The structural features represent discriminatory patterns embedded within the gradient map. Several $m \times n$ window operators on the gradient map locate local strokes in the up, down, and diagonal directions. These strokes are combined into a larger feature using a set of rules. Other features that are encapsulated include corner like shapes. When a $m \times n$ grid is used, $12 \times m \times n$ floating point numbers contribute to the total feature vector.

The coarsest set of features are the concavity features. The three types of concavity features include (i) coarse pixel density, which captures the general grouping of pixels in the image ($m \times n$ floating numbers when using a $m \times n$ grid), (ii) large strokes, which captures the prominent horizontal and vertical strokes ($m \times n \times 2$ floating numbers for a $m \times n$ grid), and (iii) up, down, left, right, and closed loop concavities detected by convolving image with a star-like operator ($m \times n \times 5$ floating numbers for a $m \times n$ grid).

4.2. Chain code and critical point features

A chain code is a lossless image representation algorithm for binary images. The basic principle of chain codes is to separately encode each connected component in the image. For each such region, the points on the boundary are traced and the

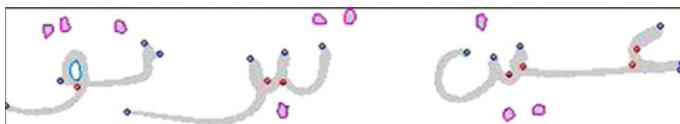


Fig. 8. Critical point features: the critical points are displayed on the image as colored pixels. Blue circles are the central locations of end critical points of a stroke, and red circles are the central locations of curvature critical points. Pink dots represent contours of small connected components, and inner loops are marked with green. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this book.)

coordinates of the boundary points are recorded. The critical point features represent three types of stroke shape features. The changes in the direction of the strokes are calculated at each local boundary point by tracing the boundary of a stroke. The change of direction at each point is estimated by measuring the angle between the incoming direction and the outgoing direction. The incoming direction and outgoing direction are estimated by using the three directly connected neighboring boundary points before and after the current point. A significant change in direction at a boundary point is determined by thresholding on the change of direction (angle) at the point. Significant left (end critical point) and right (curvature critical point) turning points in the word image are identified. These two types of points are labeled as critical points. The actual computation of these points often results in small clusters of critical points each made of several consecutive boundary points (see Fig. 8). The third type of critical points is the boundary points on small connected components, which are mostly the diacritic marks in Arabic characters.

A $m \times n$ grid is applied to the image to quantify the feature points into a feature vector of floating numbers. The ratios of the counts of each of the three types of critical points, relative to the total number of boundary points in the bin, are calculated for each of the bins. This provides three feature values in each bin and the total contribution to the feature vector is $3 \times m \times n$ floating number features.

4.3. Directional profile features

The directional profile features are inspired by the description of handwriting strokes as stroke sections in a finite number of directions, e.g., horizontal stroke, vertical stroke, etc. When a continuous stroke changes direction along the writing trajectory, the stroke is then segmented naturally into sections in different directions. Based on this concept, runlengths of black pixels are extracted along four directions, the horizontal, the vertical, and the two diagonals. Each of the text pixels is marked by a color representing the dominating direction of the runlength going through the pixel. The resultant color coded segmentation of the stroke sections can be seen in Fig. 9. The finite number of stroke directions provides a rich encapsulation of the stroke structure of the handwritten text.

A $m \times n$ grid is overlaid on the image to convert the stroke directions into a feature vector. In each of the bins, the ratio of total length of the directional runs going through each point in the bin relative to the total length of the runs going through the points in the bin is calculated. Four feature values are calculated for

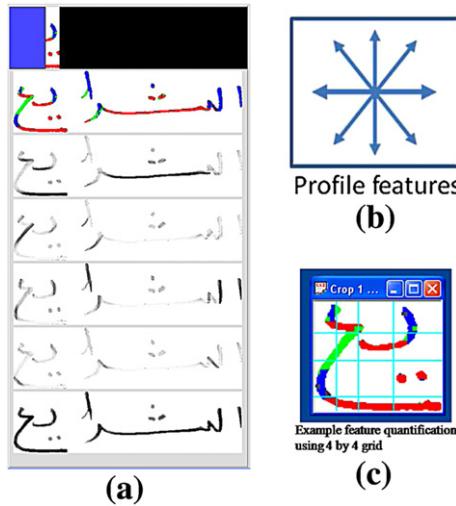


Fig. 9. (a) The first strip shows a resizable window that helps visualize the features within the window. The second strip shows the predominant directional feature at each pixel in different colors. The next four strips show the intensity of the directional features in each of the four directions (0, 45, 90, 135) with darker shades indicating greater intensity. The last strip combines the intensities of the four directions. (b) The four directions for the features (0, 45, 90, 135). (c) Example using a configurable grid of 4×4 for the PAW image.

each bin and the total contribution to the feature vector is $4 \times m \times n$ floating number features.

4.4. Discrete symbol features

Psychological studies indicate that word shape plays a significant role in visual word recognition, inspiring the use of shape-defining high-level structural features in building word recognizers (Humphreys et al., 1990; Wheeler, 1970). Ascenders and descenders are prominently shape defining and have been widely used in holistic paradigms for Latin script (Madhvanath and Govindaraju, 2001). The oscillation handwriting model (Hollerbach, 1981) theorizes that the pen moves from left to right horizontally and oscillates vertically and features near extrema are very important in character shape definition. These features include ascenders, descenders, loops, crosses, turns, and ends. Concepts such as ascender and descender are prominent when looking at a word image holistically and the position of these structures is more relevant as features than the identity of the structures themselves. So, position is an important attribute of structural features. A structural feature can also have other attributes such as orientation, curvature, and size. Structural features can be utilized together with their attributes in defining the shape of characters and, thus, the shape of words, and these features can be used to construct a recognizer that simulates the human's shape discerning capability in visual word recognition (Xue and Govindaraju, 2006).

Table 1
Structural features and their attributes

Structural feature	Position	Orientation	Angle	Width
Short cusp, long cusp	X	X		
Arc, left-ended arc, right-ended arc	X		X	
Loop, circle, cross, bar	X			
Gap				X

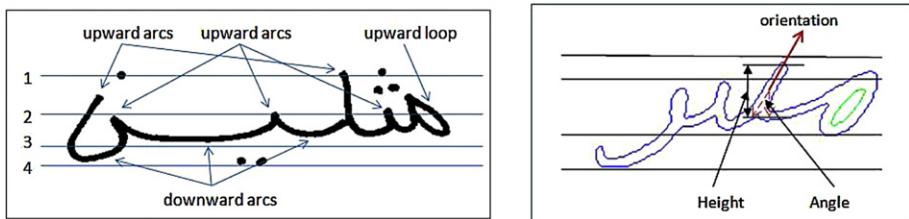


Fig. 10. Discrete symbol features.

Table 1 lists the structural features that are used in modeling handwritten characters/words. Loops, arcs, and gaps are simple features. Long cusps and short cusps are separated by thresholding their vertical length. Left/right-ended arcs are arcs whose stroke ends at its left/right side. Loops, arcs, and cusps are further divided into upward and downward classes. These features are illustrated with relation to Arabic text in Fig. 10.

4.5. Feature selection and combination

Practical systems for recognition will need to use a combination of features to obtain optimal performance. The usefulness of the features described in this section was evaluated by experiments using combinations of the implemented features.

The experiments were conducted on an image set containing 7346 PAW images of the 34 most frequent PAW classes extracted from the [AMA Arabic 1.0 Data Set \(2007\)](#). After noise removal (median filtering, slant correction, and ruling-line removal), the images were divided into a training set with 6498 PAW images and a test set with 848 PAW images. The experiments were conducted using the publicly available supporting vector machine classifier libSVM ([Chang and Lin, 2011](#)).

The best performing feature set was the combination of G and C features from the GSC feature set with the critical point and the directional profile features. Table 2 benchmarks the performance (top one recognition rate) on the 34 PAW data for each of the features and their combinations using the libSVM classifier.

The next section highlights recognition techniques proposed in the literature and discusses the issue of selection of the right model as sometimes it is difficult to approximate or locate the target concept despite sufficient number of data samples if the model selected is not appropriate.

Table 2
Performance of recognition of the 34 PAWs using libSVM (top choice)

Features	Top 1 recognition rate (%)
G in GSC	79.48
G + C	83.96
GSC	85.38
Critical point + directional profile	82.43
G + C + Critical point + directional profile	86.44

5. Models for recognition

The traditional approach for handwritten text recognition is to segment the document image into lines and further into smaller components such as characters or words. The basic units of recognition are characters or words. However, segmentation-based approaches face challenges in the case of Arabic scripts. Arabic script, be it handwritten or machine printed, is very cursive in nature. Hence, segmenting the words into characters is not easy and results in segment shapes which introduce confusion thus making recognition difficult. Arabic script has a number of dots and diacritics. Placement of these small components changes the context of the word so the same set of graphemes can create different words. Therefore, segmentation may result in loss of contextual information of the structure as a whole. The recognition models that are typically used for Arabic text recognition have words or parts of words (PAW—single connected component) as the basic unit of recognition. Some approaches that depend heavily on language models use text lines as input. This section describes techniques that have been used for Arabic handwritten word recognition.

5.1. Ensemble of learners

The task of handwritten Arabic text recognition can be seen as a complex learning problem. Dietterich (2000) lists three primary reasons for failure of learning algorithms and an ensemble can be used to address these problems. The first reason is statistical in nature in that the learning algorithm searches the hypotheses space to approximate the target concept and if the amount of training data is not sufficient to search the entire space, the algorithm outputs a hypothesis which fits the training data best. If the training data is small, there can be multiple such hypotheses with none of them not being a good approximation to the actual target concept. In such cases, the algorithm fails to learn the actual parameters of the distribution Ω and performs poorly on the test data samples. An ensemble of learners can be used to effectively address the issue of multiple consistent hypotheses and voting can be done to approximate the target concept. The second reason for failure could be that the learning algorithm may fail to reach target concept due to computational reasons. Often algorithms perform local search to optimize cost functions and get stuck in local minima like gradient descent algorithms such as Artificial Neural Networks. Likewise, decision trees greedily split the hypotheses space into half at each step and might fail to approximate or locate target concept if hypotheses space

is not smooth despite sufficient number of training data samples. Choice of a good starting point is essential for the good performance of such algorithms. Using an ensemble of learners, one can run different learners from different starting points to get a good approximation of the target concept. A third potential reason for failure is representational as it is likely that the correct target concept cannot be represented by any of the hypotheses. It could be because the learning algorithm stops searching the hypotheses space once it finds a good fit for the training samples. However by using an ensemble of learners it is possible to get a better representation of the target concept by taking weighted sum of the individual hypotheses which in turn will expand the collective hypotheses space.

[Porwal et al. \(2012\)](#) proposed a system for recognition of Arabic parts of words (PAWs) using an ensemble of biased learners in a hierarchical framework. The main motivation was that limited training data causes deterrence in searching the space and often just one kind of labeling is not enough to explore the space efficiently. Hence, an approach based on a hierarchical framework was proposed to reduce the complexity of the hypothesis space to be explored. In this method, training samples are clustered based on class labels to generate a new set of labels (according to the cluster assignment) for the data. Intuition behind doing this is to adopt a two-step approach to reduce the complexity of the task by first grouping PAWs which are similar in some way followed by a separate algorithm to distinguish the member classes within these clusters of similar PAWs. This helps in reducing the complexity of the task and the new label set will help in learning the natural structure of the data as it provides additional information. After clustering, each data point has two types of labels one corresponding to the cluster and the other one is the actual PAW class that it belongs to. A separate classifier is learned with the new set of labels generated after clustering, which will be unbiased. This is the first level of classification in the hierarchy.

In level two, an ensemble of biased learners along with an arbiter panel is used. Intuition for level two is derived from the work of [Khoussainov et al. \(2005\)](#) in which focus is on individual classes instead of different regions of instance space to find the optimal discriminant function. This can be done by constructing base learners such that they are biased toward individual classes. Learners are trained in a *one vs all* fashion where a two-class classification problem is created for each class considering the rest of the classes as the second class. To make a learner biased towards one class it is trained with majority of data points from that particular class and data points from rest of the classes as the other class. In this approach total number of classifiers trained is equal to the total number of classes. In case of disagreement between base learners, an arbiter panel is used to assign a label to the data point.

An arbiter panel is a group of classifiers with different kind of inductive biases. Inductive bias is the *assumption made by the algorithm in addition to observed data to transform its output into logical deductions* ([Mitchell, 1997](#)). There are several types of inductive biases such as maximum margin, nearest neighbor, or maximum conditional independence. The bias plays an important role when a learning algorithm outputs a hypothesis. If data points are difficult to classify even by using biased classifiers then it is intuitive to change the basic assumption made by the learner in computing the discriminant function. Hence, in the arbiter panel, learners with different inductive bias will be used to classify the data points with

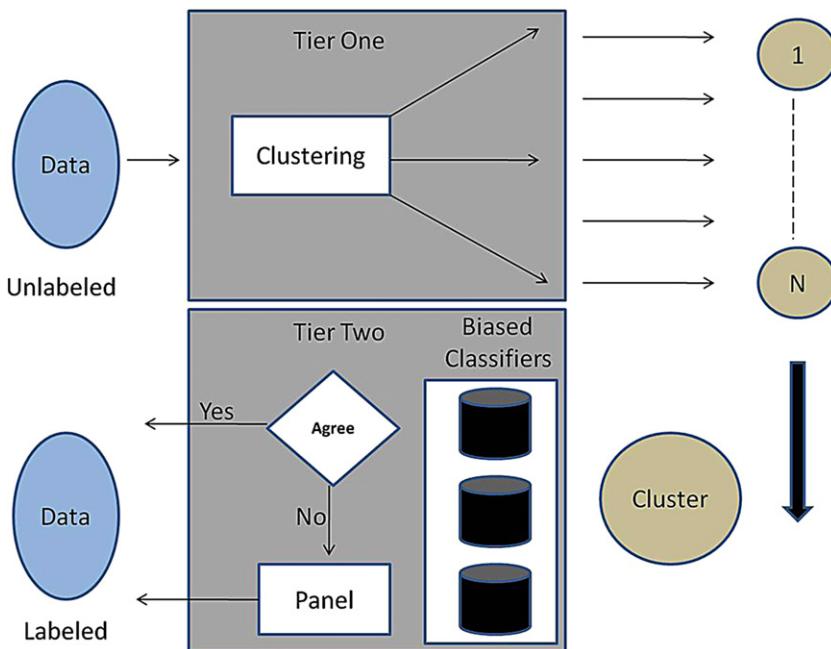


Fig. 11. Schematic of proposed biased learners-based approach in a hierarchical framework.

disagreement. An overview of the proposed system is shown in Fig. 11. Some techniques that use single classifiers instead of an ensemble of classifiers have also been successfully used for Arabic text recognition. These techniques include Artificial Neural Networks and Hidden Markov models.

5.2. Artificial Neural Network (ANN)

Artificial Neural Network (ANN) has been used extensively in various applications such as speech recognition, digit recognition, and object detection. Figure 12 (Pasero and Mesin, 2010) shows a schematic representation of an Artificial Neural Network. Fundamental working unit of an ANN is a neuron. It takes multiple inputs and generates an output \mathbf{a} which is a weighted combination of all the inputs. This output \mathbf{a} is fed into a transfer function f to produce y . An ANN is a layered collection of several such neurons as shown in Fig. 12. Weights of all the neurons are iteratively adjusted to find the minima using some constraints such as least square error. This least square error is minimized using gradient descent algorithm to find the optimal weights of the model. However, gradient descent algorithms often end up finding the local minima as opposed to global minima. Therefore, different combinations of initial weights are tried which translate to trying different starting points while exploring the hypotheses space.

Intuitively, Neural Networks change the feature representation at each level. It can be explained in terms of a complex and high dimensional target concept that the learner is seeking. When features are extracted, they hold some meaning in terms

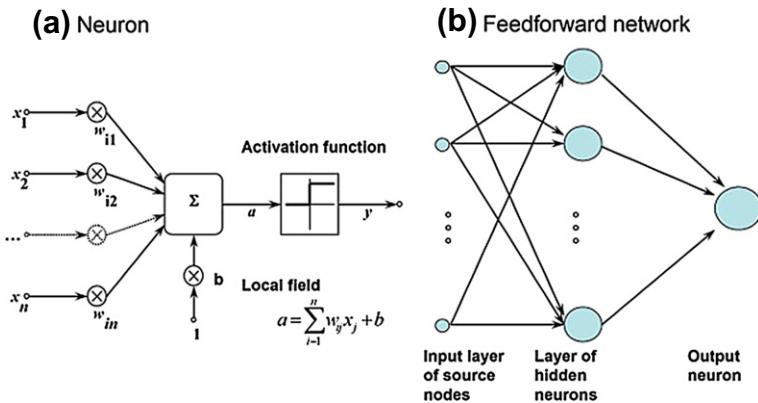


Fig. 12. Artificial Neural Network.

of a property of the image such as a structural feature or a global feature. However, it is very likely that the sought after target concept is a complex function of these features and may not be of the same dimensionality as the feature vector. Therefore, each layer of a Neural Network takes a simple representation of the data point and converts it into a more complex and abstract representation. Porwal et al. (2012) proposed a method using Deep Belief Networks (DBNs) inspired by this reasoning.

5.3. Deep Belief Networks (DBNs)

The key benefit of DBNs, which are probabilistic generative models, is to enhance the feature representation in order to approximate the target function. Primary hypothesis in the work of Porwal et al. (2012) is that the target function that could classify all the characters or words present in the lexicon will be a very high dimensional and complex function. To approximate such a function, the hypotheses space explored by the learner should also be rich enough. Using Deep Belief Networks (DBNs), simple features provided in the first layer can be mapped into more complex and abstract representation of the data. Working of DBNs is motivated by the way humans analyze and identify objects in the real world, starting from the higher level distinctions and then further breaking it down to finer details. Likewise, DBN takes features from the lowest pixel level and forms a distributed representation to discriminate at higher levels such as edges, contours, and eventually words of the handwritten text.

Any input vector \mathbf{x} can be represented in different forms (Bengio, 2009). For instance, an integer $i \in \{1, 2, \dots, N\}$ can be represented as a vector $r(i)$ of N bits where one bit corresponding to the i th position is 1 and the rest is 0. It is called *local representation*. Likewise, it can also be represented in *distributed representation* where vector $r(i)$ is of M bits where $M = \log_2^N$. It is to be noted that distributed representation is a very compact way of representation and can be exponentially more compact than the local representation. Therefore, for any input vector \mathbf{x} , $r(\mathbf{x})$ is a M way classification which partitions the \mathbf{x} space into M regions. These different partitions can be combined to form different configurations of $r(\mathbf{x})$.

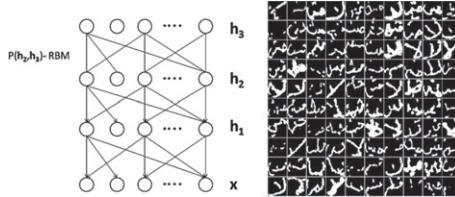


Fig. 13. Graphical model of Deep Belief Networks with one input layer and three hidden layers (left) and random samples of Arabic PAWs generated from the trained DBN.

Therefore, distributed representation is a more compact way of keeping all the information of an input feature intact. This representation is employed in deep architectures on multiple levels where higher levels are more abstract and can represent more complex forms of an input feature vector.

We have seen that structural learning can be used for feature enhancement and it is possible to increase the dimensionality of the feature space. The goal of the structural learning is to get closer to the space of the target concept irrespective of the dimensionality of features or target concept. However, it is only helpful if the classes are separable in the higher manifold. Converging on the target space may not be sufficient as it may be possible that class regions are overlapping and are not easily separable. Structural learning helps approach the space of the target concept but does not guarantee good discriminability. The information gained might be sufficient for purposes such as reconstruction or compression but if classes are not separable in that manifold, then the learning algorithm will not be able to discriminate between member classes. DBNs can be used in such settings where feature representation can be changed in each layer in such a way that discrimination occurs in a space where classes are well partitioned.

A Deep Belief Network (DBN) with one input layer and three hidden layers is shown in Fig. 13. The model is generative and the nodes in consecutive layers are fully connected, but there is no connection for nodes in the same layer. All the nodes in DBN are stochastic, latent variables. The connection between the top two layers of the DBN is undirected while the rest of the connections are directed. If a DBN has L layers then the joint probability distribution of the parameters of the generative model can be written as

$$P(\mathbf{x}, \mathbf{h}_1 \dots \mathbf{h}_L) = P(\mathbf{h}_{L-1}, \mathbf{h}_L) \left(\prod_{k=1}^{L-2} P(\mathbf{h}_k | \mathbf{h}_{k+1}) \right) P(\mathbf{x} | \mathbf{h}_1). \quad (13)$$

As can be observed from Fig. 13 the training of the directed layers (i.e., the bottom three layers) is difficult: it is intractable to obtain $p(h_1|x)$ (or $p(h_2|h_1)$) since once x is observed all the nodes in layer h_1 are correlated. However, it is relatively easy to train the undirected layer, and therefore Restricted Boltzmann Machine (RBM) is used for layer-wise pre-training.

In the pre-training phase, RBM is used to get the parameters for the first hidden layer using the input feature vector. Using the output of the first layer, second layer parameters are calculated using the RBM again. An unsupervised learning algorithm

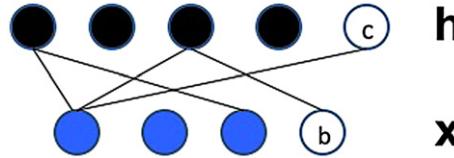


Fig. 14. Graphical model of restricted Boltzmann machines.

can unravel the salient features of the input data distribution. Once a complete network is trained, the hidden layers represent the *distributed representation* of the data and each layer is more abstract and represents higher level objects than the lower ones. Afterwards, all the parameters of the DBN are fine-tuned according to the objective criteria (Hinton and Salakhutdinov, 2006). Figure 13 illustrates samples of Arabic parts-of-words (PAWs) generated from a trained DBN.

5.3.1. Restricted Boltzmann machines (RBMs)

RBM s are used in the layerwise pre-training of the DBNs to estimate parameters for each hidden layer using the layer below it. The graphical model for RBMs is shown in Fig. 14 where all units at each layer are independent of each other. The only interaction is between the hidden layer and the observed layer.

The joint probability of binary hidden and binary observed variables can be defined by the energy function as

$$P(\mathbf{x} = \mathbf{x}, \mathbf{H} = \mathbf{h}) \propto e^{x'b + h'c + h'Wx}. \quad (14)$$

Inference in the case of RBMs is exact because the *explaining away* phenomenon is eliminated. Conditional probability of observed variable given hidden variable can be written as

$$P(\mathbf{x}_j = 1 | \mathbf{H} = \mathbf{h}) = \text{sigmoid} \left(b_j + \sum_i h_i W_{ij} \right). \quad (15)$$

Likewise, conditional probability of hidden variable given observed variable can be written as

$$Q(\mathbf{H}_i = 1 | \mathbf{x} = \mathbf{x}) = \text{sigmoid} \left(c_i + \sum_j W_{ij} x_j \right). \quad (16)$$

In training, RBMs approximate the gradient descent and the parameters can be calculated as

$$\frac{\partial \log P(\mathbf{x}; \theta)}{\partial W} = E_{P_{\text{data}}} [\mathbf{x}\mathbf{h}^T] - E_{P_{\text{model}}} [\mathbf{x}\mathbf{h}^T], \quad (17)$$

$$\frac{\partial \log P(\mathbf{x}; \theta)}{\partial b} = E_{P_{\text{data}}} [\mathbf{x}] - E_{P_{\text{model}}} [\mathbf{x}], \quad (18)$$

$$\frac{\partial \log P(\mathbf{x}; \theta)}{\partial c} = E_{P_{\text{data}}} [\mathbf{h}] - E_{P_{\text{model}}} [\mathbf{h}]. \quad (19)$$

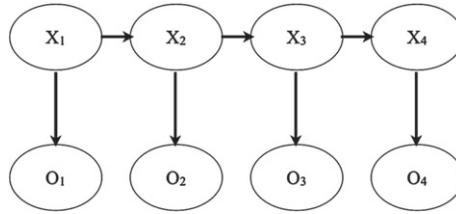


Fig. 15. Graphical representation of HMM.

In this, the expectation of the data can be calculated easily and for the expectation of the model a k -step Contrastive Divergence (CD- k) is used in which data points are sampled using Gibbs sampling (Bengio, 2009).

5.4. Hidden Markov Models (HMM)

Hidden Markov Models (HMM) have been extensively used for handwritten text recognition. Figure 15 shows a generic graphical representation of HMM where \mathbf{X} are hidden states and \mathbf{O} are the observed variables. It is based on the Markov property that any state is generated from the last few states (one in this case), therefore this is a representation of a first-order HMM. In HMM, each observation is generated by some states and observations are independent of each other. Any HMM can be defined with five parameters i.e., (N, M, A, B , and π) where N is the number of hidden states. This parameter is selected empirically and is usually based on the application and the data. M is the number of observation symbols for each hidden state. This parameter is basically the length of the observation vector. A, B , and π are learned at the time of training. Here, A is the state transition probability, B is the observation symbol probability distribution in state. This captures how any state is modeling the observed symbols. Lastly, π is the initial state probability.

In a lexicon-based approach using HMM, a separate HMM is learned for all the words in the lexicon. Features used in the training are extracted from the test word image and every HMM generates a score. Test word is identified as the word which generates the highest confidence. This is an instance of the evaluation problem in HMM where given an observation \mathbf{O} and a model λ , probability of this observation being generated by this model i.e., $P(\mathbf{O}|\lambda)$ is computed. Rabiner (1990) provides a good tutorial on the use of HMMs for recognition applications.

In Chen et al. (1995), a variable duration Hidden Markov model based approach was proposed for handwritten word recognition. In this modeling scenario, the number of states is equal to the number of characters thereby restricting the number of states. Discrete state duration probabilities were proposed as an additional element in this model. So, in addition to (N, M, A, B , and π), the model also defines D and Γ with $D = P(d|q_i)$ where q_i is any state corresponding to a letter and d is the number of segments per character. The maximum number of durations per state d was 4. The following equations summarize the calculation of the initial state probabilities.

$$\pi_i = \frac{\text{number of words beginning with } \varphi(q_i)}{\text{total number of words in the dictionary}}, \quad (20)$$

$$A_{ij} = \frac{\text{number of transitions from } \varphi(q_i) \text{ to } \varphi(q_j)}{\text{number of transitions from } \varphi(q_i)}, \quad (21)$$

$$\Gamma_j = \frac{\text{number of words ending with } \varphi(q_j)}{\text{total number of words in the dictionary}}, \quad (22)$$

$$P(d|q_i) = \frac{\text{number of times that } \varphi(q_i) \text{ is split into } d \text{ parts}}{\text{total number of times that } \varphi(q_i) \text{ appears}}, \quad (23)$$

Here, the function φ maps the state to the character.

Kim and Govindaraju (1997) used an over-segmentation approach for cursive handwritten words and the notion of variable duration obtained from segmentation statistics to maximize the efficiency of lexicon driven approaches for word recognition. The variable duration probability is used to determine the size of the matching window and improving accuracy. Character confusions were reduced by matching within the window size controlled by the statistics, the number of characters in a lexicon entry, and the number of segments of the word image.

An HMM can emit output from either transitions (Mealy machine) or states (Moore machine). **Xue and Govindaraju (2006)** describe the combination of discrete symbols and continuous attributes as structural features for cursive handwritten text recognition which are then modeled using state-emitting and transmission-emitting HMMs. This approach works very well on lexicons of size in the 100s. In all these HMM-based approaches, the Viterbi algorithm or some variation is used for decoding.

In a lexicon-free approach using HMMs, each state models one character and observation is the segmented character from the test image. This problem is an instance of the decoding problem of HMMs where given the observation sequence and a model, the sequence of states that could have generated the given observation sequence with the highest probability is computed. Since, states are characters, a word is generated by computing the optimal character sequence using dynamic programming. Although this approach does not need any lexicon, it is prone to errors as any state can go to any other state. Therefore, chances of error are high compared to a lexicon-based approach. Another limitation of this approach is that it needs characters as observations. This is a very difficult task in the context of Arabic text recognition as segmenting text into characters is hard.

6. Conclusion

In this chapter, the task of handwritten Arabic text recognition is formulated as a learning problem for a machine learning algorithm. The fundamental challenges faced by learning algorithms in general and additional challenges posed by the domain of recognition of handwritten Arabic script in particular are outlined. A discussion of the learning paradigms, features, and classification methods, that have been used for handwritten Arabic text recognition, is presented to illustrate the potential for machine learning approaches in tackling the challenging problem of unconstrained handwritten text recognition.

References

- Abney, S., 2002. Bootstrapping. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 360–367.
- Ando, R.K., Zhang, T., 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.* 6, 1817–1853. <<http://dl.acm.org/citation.cfm?id=1046920.114905>>.
- Bengio, Y., 2009. Learning deep architectures for AI. *Found. Trends Mach. Learn.* 2 (1), 1–127.
- Blitzer, J., McDonald, R., Pereira, F., 2006. Domain adaptation with structural correspondence learning. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 120–128. <<http://dl.acm.org/citation.cfm?id=1610075.1610094>>.
- Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training, In: Proceedings of the 11th Annual Conference on Computational Learning Theory, COLT' 98, ACM, New York, NY, USA, pp. 92–100. <http://doi.acm.org/10.1145/279943.279962>.
- Chang, C.-C., Lin, C.-J., 2011. LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.* 2, 27:1–27:27. <<http://www.csie.ntu.edu.tw/~cjlin/libsvm>>.
- Chen, M.-Y., Kundu, A., Srihari, S.N., 1995. Variable duration hidden markov model and morphological segmentation for handwritten word recognition. *IEEE Trans. Image Process.* 4 (12), 1675–1688.
- Clark, S., Curran, J.R., Osborne, M., 2003. Bootstrapping pos taggers using unlabelled data. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, CONLL '03, vol. 4. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 49–55. <http://dx.doi.org/10.3115/1119176.1119183>.
- Dietterich, T.G., 2000. Ensemble methods in machine learning. In: Multiple Classifier Systems, Springer, pp. 1–15.
- Favata, J.T., Srikantan, G., Srihari, S., 1994. Handprinted character/digit recognition using a multiple feature/resolution philosophy. In: International Workshop on Frontiers in Handwriting Recognition, pp. 47–56.
- Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. *Science* 313 (5786), 504–507.
- Hollerbach, J.M., 1981. An oscillation theory of handwriting. *Biol. Cybern.* 39 (2), 139–156.
- Humphreys, G.W., Evett, L.J., Quinlan, P.T., 1990. Orthographic processing in visual word identification. *Cognitive Psychol.* 22(4), 517–560. <<http://view.ncbi.nlm.nih.gov/pubmed/2253455>>.
- Applied Media Analysis, Inc., 2007. Arabic dataset 1.0. Downloaded from <<http://appliedmediaanalysis.com/Datasets.htm>>.
- Khoussainov, R., He, A., Kushmerick, N., 2005. Ensembles of biased classifiers. In: Proceedings of the 22nd International Conference on Machine Learning, ICML '05. ACM, New York, NY, USA, pp. 425–432. <http://doi.acm.org/10.1145/1102351.1102405>.
- Kim, G., Govindaraju, V., 1997. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Trans. Pattern Anal. Mach. Intell.* 19, 366–379.
- Li, S.Z., 1995. Markov Random Field Modeling in Computer Vision. Springer-Verlag, New York, Inc., Secaucus, NJ, USA.
- Liu, C.-L., Nakashima, K., Sako, H., Fujisawa, H., 2003. Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recogn.* 36 (10), 2271–2285.
- Lorigo, L.M., Govindaraju, V., 2006. Offline arabic handwriting recognition: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (5), 712–724. <http://dx.doi.org/10.1109/TPAMI.2006.102>.
- Madhvanath, S., Govindaraju, V., 2001. The role of holistic paradigms in handwritten word recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (2), 149–164. <http://dx.doi.org/10.1109/34.908966>.
- Mitchell, T.M., 1997. Machine learning. In: McGraw Hill Series in Computer Science, McGraw-Hill.
- Nigam, K., Ghani, R., 2000a. Analyzing the effectiveness and applicability of co-training. In: Proceedings of the Ninth International Conference on Information and Knowledge Management, CIKM '00, ACM, New York, NY, USA, pp. 86–93. <http://doi.acm.org/10.1145/354756.354805>.
- Nigam, K., Ghani, R., 2000b. Understanding the behavior of co-training. In: Proceedings of KDD-2000 Workshop on Text Mining.
- Pan, S.J., Yang, Q., 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* 22 (10), 1345–1359.

- Pasero, E., Mesin, L., 2010. Artificial neural networks to pollution forecast air pollution. In: InTech. <<http://www.intechopen.com/books/air-pollution/artificial-neural-networks-for-pollution-forecast>>.
- Porwal, U., Rajan, S., Govindaraju, V., 2012. An oracle-based co-training framework for writer identification in offline handwriting. In: Document Recognition and Retrieval.
- Porwal, U., Ramaiah, C., Shrivam, A., Govindaraju, V., 2012. Structural learning for writer identification in offline handwriting. In: International Conference on Frontiers in Handwriting Recognition, pp. 415–420.
- Porwal, U., Shrivam, A., Ramaiah, C., Govindaraju, V., 2012. Ensemble of biased learners for offline arabic handwriting recognition. In: Document Analysis Systems, pp. 322–326.
- Porwal, U., Zhou, Y., Govindaraju, V., 2012. Handwritten arabic text recognition using deep belief networks. In: International Conference on Pattern Recognition.
- Puurula, A., Compernolle, D., 2010. Dual stream speech recognition using articulatory syllable models. *Int. J. Speech Technol.* 13 (4), 219–230. <http://dx.doi.org/10.1007/s10772-010-9080-2>.
- Rabiner, L.R., 1990. A tutorial on hidden Markov models and selected applications in speech recognition. In: Waibel, A., Lee, K.-F. (Eds.), *Readings in Speech Recognition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 267–296. <<http://dl.acm.org/citation.cfm?id=108235.108253>>.
- Ramaiah, C., Porwal, U., Govindaraju, V., 2012. Accent detection in handwriting based on writing styles. In: Document Analysis Systems, pp. 312–316.
- Shrivam, A., Ramaiah, C., Porwal, U., Govindaraju, V., 2012. Modeling writing styles for online writer identification: a hierarchical bayesian approach. In: International Conference on Frontiers in Handwriting Recognition, pp. 385–390.
- Trier, O.D., Jain, A.K., Taxt, T., 1996. Feature extraction methods for character recognition—a survey. *Pattern Recogn.* 29 (4), 641–662.
- Valiant, L.G., 1984. A theory of the learnable. *Commun. ACM* 27 (11), 1134–1142. <http://doi.acm.org/10.1145/1968.1972>.
- Wang, W., Huang, Z., Harper, M., 2007. Semi-supervised learning for part-of-speech tagging of mandarin transcribed speech. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2007, vol. 4. pp. IV-137–IV-140.
- Wang, Y., Ji, Q., 2005. A dynamic conditional random field model for object segmentation in image sequences. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). CVPR '05, vol. 1–01, IEEE Computer Society, Washington, DC, USA, pp. 264–270. <http://dx.doi.org/10.1109/CVPR.2005.26>.
- Wheeler, D., 1970. Processes in word recognition*1. *Cognit. Psychol.* 1 (1), 59–85. [http://dx.doi.org/10.1016/0010-0285\(70\)90005-8](http://dx.doi.org/10.1016/0010-0285(70)90005-8).
- Xue, H., Govindaraju, V., 2006. Hidden Markov models combining discrete symbols and continuous attributes in handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (3), 458–462.

This page is intentionally left blank

Manifold Learning for the Shape-Based Recognition of Historical Arabic Documents

Mohamed Cheriet, Reza Farrahi Moghaddam, Ehsan Arabnejad, and Guoqiang Zhong

*Synchromedia Laboratory for Multimedia Communication in Telepresence,
École de technologie supérieure, Montreal, QC, Canada H3C 1K3*

Abstract

In this work, a recognition approach applicable at the letter block (subword) level for Arabic manuscripts is introduced. The approach starts with the binary images of the letter block to build their input representation, which makes it highly objective and independent of the designer. Then, using two different manifold learning techniques, the representations are reduced and learned. In order to decrease the computational complexity, PCA is applied to the input representations before manifold learning is applied. Also, in order to increase the performance and quality of the input representations, a gray stroke map (GSM) is considered in addition to the binary images. The performance of the approach is tested against a database from a historical Arabic manuscript with promising results.

Keywords: document image processing, document image understanding, optical shape recognition, manifold learning

1. Introduction

The recognition of text on historical manuscripts is of great interest around the world for understanding the huge volumes of digitized manuscripts that have been produced (Antonacopoulos and Downton, 2007; Abid, 1997; Barni et al., 2008). The availability of live text not only provides the ability to perform online searching in the manuscript, it boosts the research conducted in the field of history and philosophy by an order of magnitude, and this will have a vast social and environmental impact on the societies involved. However, the recognition and extraction of text from historical manuscripts are not an easy task, considering the wide variations in writing styles, including writer-related variations, as well as degradation. In cases where recognition

is not possible, other solutions, such as word spotting, have been employed to create indices of manuscripts (Farrahi Moghaddam and Cheriet, 2009a; Rath and Manmatha, 2007; Ball et al., 2006).

The situation is a great deal worse for manuscripts in Arabic (Lorigo and Govindaraju, 2006). This is partly because Arabic script is cursive by nature, which makes it difficult for many of the recognition techniques developed for Latin script to work with it. Also, there are many different writing styles for Arabic script, such as Naskh, Nastaliq, etc., which are not only different in terms of character strokes, they also follow completely different rules of calligraphy (Milo, 2009, 2010). For example, the baselines in Nastaliq Arabic, which is a Persian style, are not horizontal lines; moreover, they shift up and down along the text line (Wali and Hussain, 2007). In many cases, there are not enough pixels associated with a letter in a letter block for it to be learned. Also, the imprecise placement of diacritics and dots, especially dots, is a common issue in Arabic styles. Some samples of Nastaliq script are provided in Fig. 1 which show aforementioned difficulties, including not horizontal lines and not sequential appearance of letters. The images are courtesy of Nastaliq Script Exhibition,¹ Kakayi: The Art of Oriental Calligraphy,² and Hamid Akbari Gallery.³

Recognition is a highly multi-class classification problem at the word level (meaningful level). In the Arabic language, there are more than 280,000 unique words (AbdelRaouf et al., 2010). By moving to the lower level of the letter blocks (subwords or connected components), the number of unique classes drops to 66,000, which reduces the complexity of the classification. It is worth noting that recognition systems usually work at a much lower (character) level in the case of Latin script, which drastically reduces the number of classes to 26. However, other approaches should be considered because in many old manuscripts segmentation of subword images into individual characters, graphemes, or even vertical lines is very difficult because of not sequential nature of their script and lack of presence of a unique baseline (see Fig. 1, for example). As the classification at a scale of 60,000 classes is still an unsolvable problem, other approaches could be pursued in order to reduce the number of classes at the letter block level, for example, the binary descriptors approach (Farrahi Moghaddam et al., 2010, 2012). In the binary descriptors approach, which we will use in this work, the class definition is kept at the character level, but the samples (observations) are moved to the letter block level (Farrahi Moghaddam et al., 2010, 2012). In other words, each object (character) is separately learned, in the form of a binary descriptor, on the samples (the letter blocks), and then all the binary descriptors are combined at the end to retrieve the letters of a letter block. Usually, a skeleton-based or curve-based representation of the letter blocks is used in order to reduce their complexity. It has been observed that a few hundreds of binary descriptors are sufficient to learn all possible subwords (Farrahi Moghaddam et al., 2012). These descriptors include the position of a few characters at the beginning of letter blocks in order to reconstruct the whole letter sequence of the letter block. In other words, in the binary descriptors approach,

¹ <http://calligraphy.blogfa.com/8505.aspx>.

² <http://kakayicalligraphy.webs.com/shekastastyle.htm>.

³ <http://payameghalam.persianblog.ir/post/1139>.



Fig. 1. Some samples of the Nastaliq script: (a and b) Courtesy of Nastaliq Script Exhibition. (c) Courtesy of Kakayi: The Art of Oriental Calligraphy. (d) Courtesy of Hamid Akbari gallery.

instead of trying to segment the subword image into character/letter sub images, a set of binary descriptors is defined, which, if learned, could be used to reconstruct the associated string of subwords (Farrahi Moghaddam et al., 2010, 2012). Some examples of binary descriptors are: (i) the presence of a specific letter in the subword (regardless of its position); (ii) the presence of a specific letter as the first letter of the subword; (iii) the presence of a specific letter as the second letter of the subword; (iv) more than one occurrence of a letter in a subword; and (v) the number of letters in a subword.

It is worth noting that subwords processing can also be used for word spotting (Cheriet and Farrahi Moghaddam, 2012). As retrieval of the live text is not required in the word spotting applications, no explicit learning is needed, and usually a clustering process can provide spotted instances of a query image based on its shape (subword) level features.

As the learning part of the system is almost independent of the class management part, we ignore the latter and focus on the learning part, which is a difficult problem in itself. The factors contributing to this learning difficulty are: the variability of the domain (different sizes of letter block images), the wide spectrum of complexity in shapes, the lack of upper and lower profiles because of backward strokes, and non horizontal baselines, among many others.

The most important stage in a learning process is the generation and selection of the appropriate features for representing the objects in an optimal way, i.e., obtaining the maximum amount of information about the problem, while at the same time reducing the noise in the representation. Subword (word) images are rich in visual information, and this makes feature selection more difficult. Usually, a fraction of the information available is extracted and used as features. Skeleton images, in contrast, contain much less noisy information (Mahmoud et al., 1991; Steinherz et al., 2000). However, it is much more difficult to compare them at the image level, and so it is the high-level features that are usually extracted from them (Zhu, 2007). For example, in Farrahi Moghaddam et al. (2010), several topological and geometrical descriptors of variable dimensions have been used to represent a skeleton. Using the contour of the subword is another means for vectorizing subword images (Belongie et al., 2002; Mahmoud, 1994). Other approaches include using pixel density and a histogram of directions (Mezghani et al., 2008). In all these approaches, the features are extracted or reduced based on the subjective view of the model designer. Although all these approaches and their combinations will eventually be improved and converge to produce the optimal set of features in the future, a more objective way to achieve this would be direct feature reduction. It is, in fact, critical to do this in many applications, where there is no cue or hint as to the way in which to represent the objects in raw form, and the number of features is very high. Examples include genome projects, text categorization, image retrieval, and customer relationship management (Yu et al., 2003). In this work, we apply various approaches to reduce the number of input features, which are constructed from the normalized binary image of each subword and one of its generated maps.

Various approaches have been used to reduce the number of features. Both unsupervised methods, such as PCA (Pearson, 1901) and ICA (Hyvarinen and Oja, 2000), and supervised methods, such as LDA (Fisher, 1936; Radhakrishna Rao, 1948), have been used for this purpose. All these methods assume a predefined linear or nonlinear model for the data. To arrive at a more objective approach, we will use manifold learning methods (Baudat and Anouar, 2000; Geng et al., 2005) to rearrange the representations and reduce the number of features. Details of the manifold learning approaches are presented in Section 3.

In this work, a shape-based recognition of Arabic letter blocks (subwords) is investigated. Raw pixel-level features from the binary images and the gray stroke map (GSM) of letter blocks are considered as the initial representation. Then, two different manifold learning approaches, GDA and LLE, are used to reduce the complexity of the representation. The performance of the proposed approach is evaluated against a database of letter blocks from a real manuscript (Wisnovsky, 2004).

The paper is organized as follows. In Section 2, the problem statement is provided. A brief description of two manifold learning approaches is presented in Section 3. A description of the features extracted from the binary images and the GSM are provided in Section 4. Experimental results are discussed in Section 5. In Section 6, our conclusion and some prospects for future work are presented.

2. Problem statement

A ground truth collection of letter block images (a letter block being a series of letters and their associated diacritics and dots) of a writing style of Arabic script is available. Each letter block is associated with its string. The goal is to build a system that can obtain the text of each letter block of that writing style. The system will use the pixel-level features of binary images and a GSMs of the letter blocks. Several manifold learning approaches are used.

3. Manifold learning

Dimensionality reduction is an essential step in high-dimensional data analysis. The dimension reduction algorithms are applied before the classification algorithms, as a data preprocessing step, in order to arrive at a minimal number of features by removing irrelevant, redundant, and noisy information. Large amounts of data, such as the image data, are considered as high-dimensional, and most recognition systems use a linear method that ignores the properties of manifolds. Nonlinear dimensionality reduction methods are commonly used for two purposes in reducing the number of input variables; for extracting the manifold of the most important features and organizing the data for better visualization. Below, two different approaches to manifold learning are discussed.

3.1. Locally linear embedding (LLE)

Locally linear embedding (LLE) (Roweis and Saul, 2000) makes the assumption that each point can be reconstructed locally by its neighbors, and the low-dimensional representation of data can be achieved using reconstruction weights. The process can be summarized in the following three LLE steps:

- (1) For each data point, the distances between that point and the others are computed, and its K -nearest neighbors are selected.
- (2) The reconstruction weights of all points are calculated using K -NN:

$$\widetilde{W} = \arg \min_W \epsilon(W) = \arg \min_W \sum_i \left\| x^i - \sum_j w_{ij} x^j \right\|^2 \quad \text{s.t. } \sum_j w_{ij} = 1, \quad (1)$$

where \widetilde{W} is the calculated weight matrix.

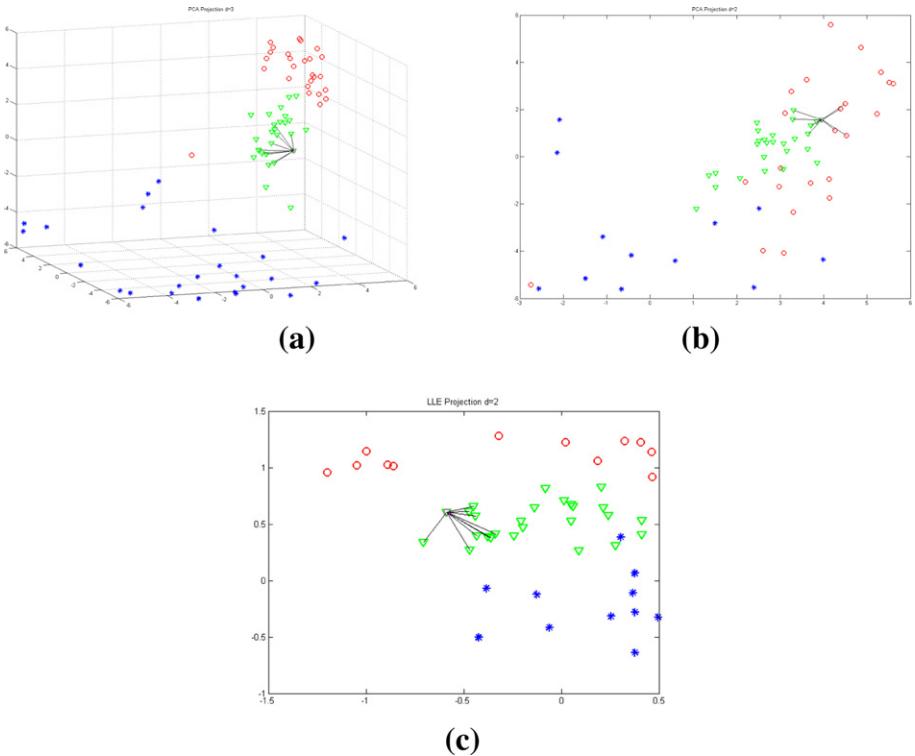


Fig. 2. An example of dimension reduction to two dimensions using LLE. (a) The original data. (b) Reduction to 2D using PCA. (c) Reduction to 2D using LLE.

(3) The embedding coordinates are computed by minimizing the reconstruction error ϕ of the coordinates using W :

$$\tilde{Y} = \arg \min_Y \phi(Y) = \arg \min_i \sum_y \left\| Y^i - \sum_j w_{ij} Y^j \right\|^2, \quad (2)$$

where \tilde{Y} is the embedded representation. A typical example of LLE is shown in Fig. 2. Figure 2a shows the original data. For the sake of presentation, the original data reduced to three dimensions using PCA is shown this figure. Figure 2b shows the reduction to two dimensions using PCA. As it expected, the overlapping between-classes is very high. In contrast, Fig. 2c shows the reduction obtained using the LLE approach. The overlapping is much less, and is expected to be lower when supervised LLE is used.

3.2. Supervised locally linear embedding (SLLE)

LLE is not directly applicable into many pattern recognition problems because it does not consider the known class label information of the input data. The purpose of introducing SLLE (supervised locally linear embedding) is to use special mappings that separate a within-class structure from a between-class structure. One

approach to doing this is to add a term to the distance between samples from different classes which only modifies the first step of the original LLE, and leaves the other two steps unchanged. This can be achieved by artificially increasing the pre-calculated Euclidean distance between samples from different classes, but leaving these distances unchanged if the samples are from the same class (Geng et al., 2005):

$$D' = D_{\text{Euc}} + \alpha \max(D_{ij})(1 - \delta_{ij}), \quad (3)$$

where δ_{ij} is Kronecker's delta.

Another method of imposing supervision on LLE is to apply some shrink/expand functions which decrease the in-class distance while increasing the between-class distance (Zhang, 2009):

$$D_{ij} = \begin{cases} \sqrt{e^{D_{ij}^2/\beta} - \alpha}, & \text{if } i \neq j; \\ \sqrt{1 - e^{-D_{ij}^2/\beta}}, & \text{if } i = j. \end{cases} \quad (4)$$

The parameter β is set to the average Euclidean distance of all training data, and the values of the parameters α and K are set in an optimizing procedure to get a minimum error rate in the training set.

The above algorithms only change the first step of the original LLE algorithm. We will use the new distance (4) in our experiments.

LLE does not provide a straightforward method for the embedding of new data that are not in the training set. This is also the case for the supervised LLE. The supervised LLE only projects the training data, not the test data. In order to apply the learned manifold to a new data point in the feature space, we follow the non-parametric kernel-based approach using the regression weights matrix introduced in Bengio and Vincent (2003).

3.3. A measure of manifold quality

The only parameters that should be determined by a human in LLE are the number of nearest neighbors and the embedding dimension. The embedding dimension depends on the number of nearest neighbors. However, because of the nature of the LLE algorithm, we cannot select an embedding dimension larger than the number of nearest neighbors. Selecting K is a challenge in LLE, because, with a small K , we lose at the global scale, and with a large K we lose at the local scale. So, we must have some criteria to determine the number of nearest neighbors in an optimal way.

In Kouropeteva et al. (2002), the residual variance is used to measure the embedding quality:

$$\sigma_R^2 = 1 - \rho_{D_X D_Y}^2, \quad (5)$$

where ρ^2 is the correlation of the distance matrix in the original space and the embedding space. The minimum value for the residual value (sum of the squared errors) corresponds to the best representation.

In Valencia-Aguirre et al. (2009), a measure of the embedding quality is introduced as follows:

$$C(X, Y) = \frac{1}{2n} \sum_{i=1}^n \left\{ \frac{1}{n} \sum_{j=1}^n \{D(x_i, \eta_j) - D(y_i, \eta_j)\}^2 + \frac{1}{k_n} \sum_{j=1}^{k_n} \{D(x_i, \theta_j) - D(y_i, \gamma_j)\}^2 \right\}. \quad (6)$$

The first term represents the local properties in the embedding and high-dimensional spaces, and this quantitative measure illustrates how well the distance information is preserved. The second term represents the error that occurs when the points far away in the high-dimensional space are mapped close together in the embedding space, owing to the selection of the incorrect number of nearest neighbors. The procedure for selecting the optimal number of neighbors is as follows:

- Start the LLE algorithm with the initial K and compute the embedding.
- Calculate the embedding quality using the selected criteria.
- Change K until the minimum value of C is achieved that corresponds to the optimal K .

3.4. Generalized discriminant analysis

Generalized discriminant analysis (GDA) (Baudat and Anouar, 2000) is a kernelized variant of linear discriminant analysis (LDA) (Fisher, 1936). However, unlike LDA, which seeks a *linear* projection that simultaneously minimizes the within-class scatter and maximizes the between-class scatter to separate the classes, GDA pursues a *nonlinear* mapping. Hence, GDA overcomes the limitation of LDA that it can only deliver linear projection of the data.

LDA is a commonly used statistical approach for dimensionality reduction. Suppose we are given N training data, $\mathcal{T} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbb{R}^D, i = 1, 2, \dots, N$. We denote the number of classes as C , and the number of samples in class Δ_j as $n_j, j = 1, 2, \dots, C$. The “between-classes scatter matrix,” \mathbf{S}_B , and the “within-classes scatter matrix,” \mathbf{S}_W , are defined as

$$\mathbf{S}_B = \sum_j n_j (\mu_j - \mu)(\mu_j - \mu)^T, \quad (7)$$

and

$$\mathbf{S}_W = \sum_j \sum_{\mathbf{x}_i \in \Delta_j} (\mathbf{x}_i - \mu_j)(\mathbf{x}_i - \mu_j)^T, \quad (8)$$

where

$$\mu = \frac{1}{N} \sum_i \mathbf{x}_i, \quad (9)$$

and

$$\mu_j = \frac{1}{n_j} \sum_{\mathbf{x}_k \in \Delta_j} \mathbf{x}_k, \quad (10)$$

are the overall mean of the training data and mean of samples in class Δ_j , respectively. To find the optimum projection matrix, \mathbf{P} , LDA maximizes the following objective:

$$J(\mathbf{P}) = \frac{\text{tr}(\mathbf{P}^T \mathbf{S}_B \mathbf{P})}{\text{tr}(\mathbf{P}^T \mathbf{S}_W \mathbf{P})}, \quad (11)$$

where $\text{tr}(\cdot)$ is the trace of a square matrix. Since J is invariant with respect to (w.r.t.) the rescalings of \mathbf{P} : $\mathbf{P} \rightsquigarrow \eta \mathbf{P}$, we can enforce the denominator to be simply $\text{tr}(\mathbf{P}^T \mathbf{S}_W \mathbf{P}) = 1$. To the end, we can rewrite the problem of maximizing J into a constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{P}} \quad & -\text{tr}(\mathbf{P}^T \mathbf{S}_B \mathbf{P}) \\ \text{s.t.} \quad & \text{tr}(\mathbf{P}^T \mathbf{S}_W \mathbf{P}) = 1. \end{aligned} \quad (12)$$

Introducing the Lagrangian multiplier, λ , the Lagrange function w.r.t. Problem (12) can be written as

$$\mathcal{L}(\lambda, \mathbf{P}) = -\text{tr}(\mathbf{P}^T \mathbf{S}_B \mathbf{P}) + \lambda(\text{tr}(\mathbf{P}^T \mathbf{S}_W \mathbf{P}) - 1). \quad (13)$$

Minimizing Problem (13) is equivalent to solving a generalized eigenvalue decomposition problem:

$$\mathbf{S}_B \mathbf{P} = \mathbf{S}_W \Lambda \mathbf{P}, \quad (14)$$

where Λ is a diagonal matrix. Since the rank of \mathbf{S}_B is at most $C - 1$, the obtained solution that maximizes J generally includes $C - 1$ eigenvectors of Eq. (14) corresponding to the nonzero eigenvalues.

Using the so-called kernel trick, GDA adapts LDA to its nonlinear version. The main idea of GDA is to map the input space into a high-dimensional (possibly infinite) feature space in which variables are nonlinearly related to the input space. This technique has also been applied to some other algorithms, such as kernel principal component analysis (KPCA) (Schölkopf et al., 1998) and support vector machines (SVMs) (Vapnik, 1995, 1998). Let $\phi(\cdot)$ denote the nonlinear mapping from the data space, \Re^D , to the reproducing kernel Hilbert space (RKHS), \mathbf{H} , which corresponds to a kernel function $k(\cdot, \cdot) = \phi(\cdot)^T \phi(\cdot)$. The “total scatter matrix,” \mathbf{S}_t , and the “between-classes scatter matrix,” \mathbf{S}_b , are defined as

$$\mathbf{S}_t = \sum_i (\phi(\mathbf{x}_i) - \mathbf{m})(\phi(\mathbf{x}_i) - \mathbf{m})^T, \quad (15)$$

and

$$\mathbf{S}_b = \sum_j n_j (\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T, \quad (16)$$

where

$$\mathbf{m} = \frac{1}{N} \sum_i \phi(\mathbf{x}_i), \quad (17)$$

and

$$\mathbf{m}_j = \frac{1}{n_j} \sum_{\mathbf{x}_k \in \Delta_j} \phi(\mathbf{x}_k), \quad (18)$$

are the overall mean of the data in the feature space \mathbf{H} and class mean of Δ_j , respectively.

To implement LDA in the feature space \mathbf{H} , GDA optimizes the following trace function w.r.t. \mathbf{S}_t and \mathbf{S}_b :

$$\mathbf{W}^* = \arg \min_{\mathbf{W}} \text{tr}((\mathbf{W}^T \mathbf{S}_t \mathbf{W})^{-1} \mathbf{W}^T \mathbf{S}_b \mathbf{W}). \quad (19)$$

However, since the explicit form of $\phi(\cdot)$ is unknown, Problem (19) can not be straightforwardly solved via generalized eigenvalue decomposition. Fortunately, according to the representer theorem (Kimeldorf and Wahba, 1971; Schölkopf and Smola, 2002):

$$\mathbf{W} = \mathbf{XP}, \quad (20)$$

where $\mathbf{X} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \dots, \phi(\mathbf{x}_N)]$ is the data matrix in the feature space, \mathbf{H} , and \mathbf{P} is the coefficient matrix. Thus, Problem (19) can be rewritten as

$$\mathbf{P}^* = \arg \min_{\mathbf{P}} \text{tr}((\mathbf{P}^T \mathbf{K} \mathbf{H}_N \mathbf{K} \mathbf{P})^{-1} (\mathbf{P}^T \mathbf{K} \mathbf{H}_N \mathbf{G} \mathbf{H}_N \mathbf{K} \mathbf{P})), \quad (21)$$

where \mathbf{K} is the kernel matrix, $\mathbf{H}_N = \mathbf{I}_N - \frac{1}{N} \mathbf{1}_N \mathbf{1}^T$ (\mathbf{I}_N is the $N \times N$ identity matrix and $\mathbf{1}_N$ is an $N \times 1$ vector of all ones), and \mathbf{G} is a similarity matrix defined as

$$\mathbf{G}(s, t) = \begin{cases} \frac{1}{n_j}, & \mathbf{x}_s \in \Delta_j \text{ and } \mathbf{x}_t \in \Delta_j; \\ 0, & \text{otherwise.} \end{cases} \quad (22)$$

Similar to LDA, the optimal solution \mathbf{P}^* of Problem (21) generally includes $C - 1$ eigenvectors of $(\mathbf{K} \mathbf{H}_N \mathbf{K})^{-1} (\mathbf{K} \mathbf{H}_N \mathbf{G} \mathbf{H}_N \mathbf{K})$ corresponding to the nonzero eigenvalues.

4. Feature extraction

As has been mentioned, we use the raw binary image of a letter block as the input representation and features. The color document images of the manuscript are first



Fig. 3. Two examples of letter blocks of different complexities.

binarized using the grid-based Sauvola method (Farrahi Moghaddam and Cheriet, 2010). After attaching the diacritics and dots to their hosting connected components (CCs), the binary images of letter blocks are extracted. Two examples of a letter block are shown in Fig. 3. As will be discussed in Section 5, padding and translation are used to make the binary images the same size. As Arabic script is written from right to left, padding is performed on the left side of the images. Also, the letter block images are centered vertically on their baseline, which is calculated using a modified local averaging on its neighboring CCs.

4.1. Gray stroke map (GSM)

The stroke map (SM) was introduced in Farrahi Moghaddam and Cheriet (2009b). The main concept behind SM is the identification of pixels of interest (POI) that have a stroke structure around them. The SM is a map which assigns a value to each pixel on the input image. In the SM, the finite and fixed width of the pen, which is measured in the form of the average stroke width w_s (Farrahi Moghaddam and Cheriet, 2010), is the main clue for identifying the stroke pixels. Various implementations of SM have been used from the kernel-based methods (Farrahi Moghaddam and Cheriet, 2009b; Ye et al., 2001) to overlap patches (Farrahi Moghaddam and Cheriet, 2010). In this work, another implementation of SM, based on the skeleton image and w_s , is proposed, in order to reduce the computational time. The details of the implementation are provided in Algorithm 1. One of intermediate states in the computation of the SM in this implementation is defined as a new map: the gray stroke map (GSM). This map gives the probability (membership value) of a pixel belonging to the text strokes. An example illustrating this is given in Figs. 4 and 5. A rough and oversegmented binarization is corrected in the GSM and SM results, thanks to the w_s *a priori* information. In this work, GSM is used as one of the two-dimensional maps.

5. Experimental results

For the experiment, a database of 2400 images of 60 unique subwords is selected from a real Arabic manuscript which was used in the construction of the IBN SINA

Algorithm 1: Estimation of SM and GSM:

- 1 Get the input document image, its rough binarization, and *a priori* information including w_s and h_l ;
- 2 Estimate the edge map of the document image at the h_l scale (using Sobel's method (Sobel and Feldman, 1968), and mosaicking the image with squares of size $h_l \times h_l$);
- 3 Estimate the skeleton map of the document image using the thinning method;
- 4 Ignore the skeleton whose distance to the edges is more than $[w_s/2] + 1$;
- 5 Produce the Euclidean distance map of the skeleton map capped to the $w_s + 1$;
- 6 Estimate the text edges based on the calculated distance map (by selecting those pixels which have a value of $w_s + 1$ on the distance map and are presented on the edge map of Step 2);
- 7 Produce the Euclidean distance map of the new edge map capped to the $w_s + 1$;
- 8 Combine the two distance maps, obtained in Steps 5 and 7, to generate the gray stroke map (GSM);
- 9 Threshold the GSM on 0.5 to generate the stroke map (SM);

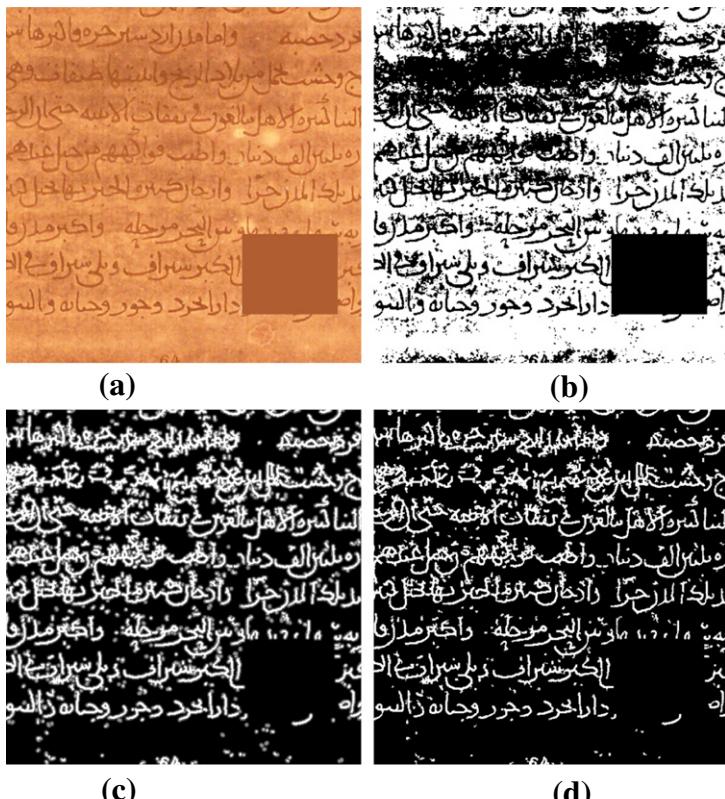


Fig. 4. Illustration of SM performance. (a) Input image suffering from degraded background and synthetic degradation. (b) Rough binarization of the (a) using Otsu's method. (c and d) GSM and SM of (b) generated using Algorithm 1.

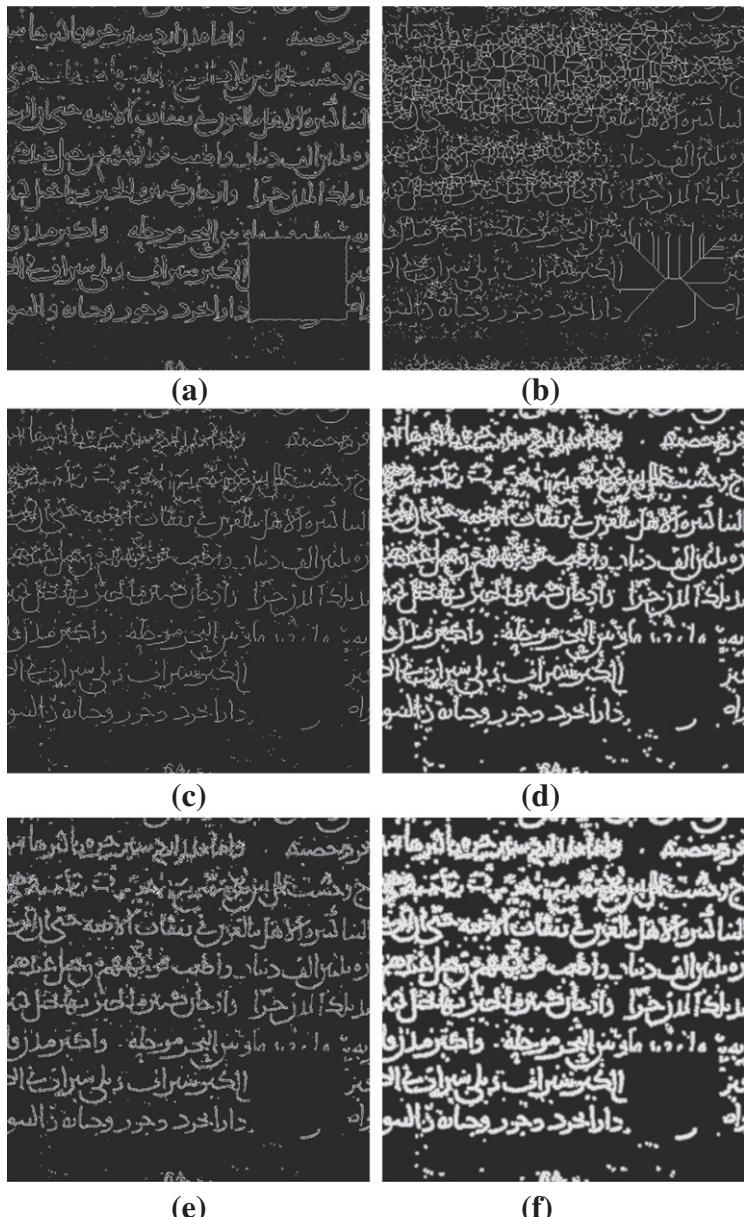


Fig. 5. The detailed steps used to produce the outputs in Fig. 4. (a) The edge map calculated in Step 2 of Algorithm 1. (b) The skeleton map calculated in Step 3. (c) The corrected skeleton map calculated in Step 4. (d) The Euclidean distance map of the skeleton map as calculated in Step 5. (e) The estimated text edges calculated in Step 6. (f) The Euclidean distance map of the new edge map as calculated in Step 7.

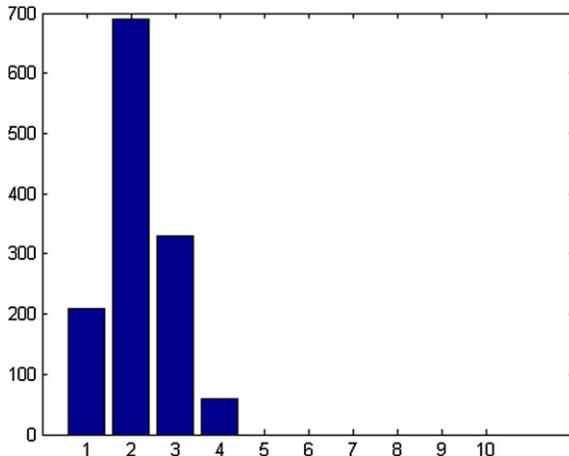


Fig. 6. Histogram of the image letter blocks based on label length.

database (Farrahi Moghaddam et al., 2010). It is worth noting that the number of binary descriptors is independent from the number of sample. For example, if we just need to consider the descriptors which count the presence of a character in a letter block, we will have only 28 binary descriptors presented. However, in this work, because of small size of the database, instead of the binary descriptor, the actual labels of the 60 samples are considered as the binary labels to be learned. We will use larger databases along with actual binary descriptors in the future work. In Fig. 6, the histogram of the number of letters in the database letter blocks is shown. The subword images are resized using padding; in the first step (resizing step), the image with the maximum width is selected and the other images are resized by adding zero/one columns (based on the background being zero/one) to the left or right side of the images. Next, the image with the maximum height is selected, and the other images are resized by adding zero/one row to the top or bottom of the image based on the baseline. Because Arabic is written from right to left, we add zero columns to the left side of the images. Each image is converted to a feature vector of $1 \times N$, and the database is created by concatenating all the feature vectors. In our case, each feature vector has 13,480 elements.

For each experiment, we use two cases to create the database: (i) binary images (BIN) and (ii) binary and GSM images (BIN+GSM). This means that, for each sample, we use the binary image and convert it to a feature vector in one case, and we use the binary image and the GSM image and convert them to a vector in the second case, creating the final feature vector by concatenating them.

5.1. PCA

In the first experiment, PCA is applied for dimension reduction, and then a K -NN classifier is applied. The result constitutes our baseline for comparing performances. In the first step, the mean and covariance matrices of the features are computed, and the principal components are calculated using eigenvalue decomposition of

Table 1

The Experimental results of the PCA and K -NN classification case

	Data type	k	Training set (%)	Testing set (%)
1	Binary	1	76.8	72.2
2	Binary+GSM	1	80.1	75.9

Table 2

The Experimental results of the PCA and RF classification case

	Data type	Dimensions	Number of trees	Testing set (%)
1	Binary	1200	460	78.61
2	Binary	256	420	81.25
3	Binary+GSM	1200	440	80.33
4	Binary + GSM	256	440	82.97

the covariance matrix. The eigenvalues are sorted and the n maximum values are selected. The corresponding eigenvectors are used for the projection. The value of n can be selected based on the energy of the components. For example, this value could be selected so that 90% of the energy is preserved. In our database, the data dimension is 13,480 and we kept 1200 eigenvectors for the projection in order to preserve more than 99% of the energy. The K -NN classifier is first applied on training and verification data to select the optimal K , which is then used in the testing step. Table 1 shows the result of applying the first experiment with the two types of data, BIN and BIN+GSM. It is worth noting that, in the second case, in which GSM is used in the input representations, the performance is improved by 3%.

Also, as a second baseline, the ensemble learning using Random Forest technique (Breiman, 2001) is applied to the database. The results are presented in Table 2.

5.2. PCA+GDA

In the second experiment, we use the GDA algorithm in the dimension reduction step, in addition to PCA. The RBF kernel is used in the GDA. In order to select the best value for the parameter σ , the data is divided into two parts: training and verification. At each step, the GDA model is created using the training database and the parameter is selected, and then we test the model using the verification set. We select the parameter that maximizes the rate of classification. In Table 3, the results of the second experiment are shown. The results are obtained using the RBF kernel with $\sigma = 11$. As can be seen from the results, the application of manifold learning has increased the performance by 5%.

5.3. PCA+LLE

In the third experiment, we use the LLE algorithm as our dimensionality reduction approach. The Euclidean distances between all the samples are calculated and the

Table 3
The Experimental results of the PCA+GDA approach

	Data type	k	Training set (%)	Testing set (%)
1	Binary	1	80.4	77.3
2	Binary+GSM	1	85.4	81.3

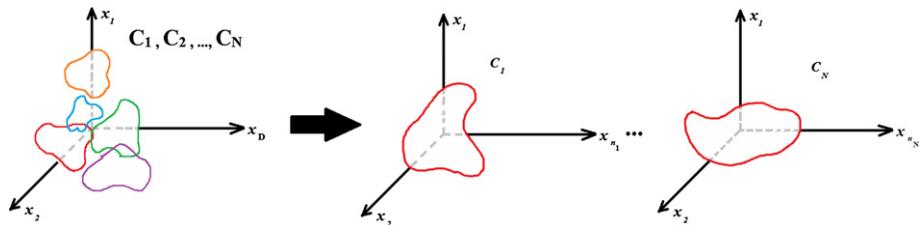


Fig. 7. Using label-dependent distance, the data are separated into several manifolds, one for each class.

distance matrix is modified by applying the distance that is used in SLLE, the distance (4). After this step, the samples with the same labels become closer, and the samples with the different labels grow farther apart. We applied a clustering algorithm to divide the samples into C clusters, as shown in Fig. 7. In this experiment, we know the number of classes; therefore, we set a value of 60 to the number of clusters.

Using the modified distance matrix, we expect the samples with the same labels to be clustered. A manifold for each cluster (we have 60 clusters here) is created using the LLE algorithm. The Euclidean distance of the samples is used to select the K -nearest neighbors, and then the weights of reconstruction of each sample by its neighbors are calculated. The eigenvalues and eigenvectors of the cost matrix $M = (I - W)'(I - W)$ are calculated, and the samples are projected onto a low-dimensional space constructed using d eigenvectors, corresponding to the d smallest eigenvalues, $d = K - 1$ in this experiment. The quality of the projection can be analyzed by some criterion, such as (5) or (6). In this experiment, we selected the second criterion, which checks whether or not the local property of the data in the original space is preserved correctly in the projected space. Ideally, the samples should have the same nearest neighbors in the original and projected spaces. This means that samples which are close in the original space should be close in the projected space, and samples which are far apart in the original space should be far apart in the projected space. K is the only parameter that exists in LLE that affects the property mentioned above. So, the optimum value of K can be obtained by changing the value of K and projecting the data, and then measuring the manifold quality for achieving the optimum value that minimizes the criterion. The samples in the training set are used to optimize the parameter, and the samples in the testing set are used to measure the performance of the algorithm. Optimal K values for some of the manifolds are presented in Table 4. The learning steps of this experiment are as follows:

- Select an initial value for K .
- Create a manifold (projecting the samples) using the selected K .
- Measure the quality of the manifold by means of criterion (6).
- Change the value of K and repeat the two previous steps.
- Find the optimum value for K .
- Repeat the previous steps for each cluster to find the best parameter.

In the testing procedure, each new piece of data should be projected onto all the manifolds, so that the decision will be the label of the new data, which is the label of one of these manifolds. In the learning step, as we create each manifold separately and because the optimum values of K are different for each manifold, the direct combination of the result of the projection onto different manifolds is not possible. The idea behind LLE is to project data into the new space while preserving the local properties of the data, and this is achieved by finding the weights of reconstruction of one sample by its nearest neighbors. So, if the new data are projected into the proper manifold, there should be a minimum of reconstruction errors. The testing steps are as follows (see Fig. 8):

- Find the K -nearest neighbors of a new sample, x_t , on each manifold, say the i th manifold, using the Euclidean distance and the optimum value of K for that manifold.
- Find the reconstruction weights in the i th manifold: w_j .
- Calculate the reconstruction error in the i th manifold:

$$e_i(x_t) = \left\| x_t - \sum_j w_j x_j \right\|. \quad (23)$$

Table 4

Optimum value of K for some sample subwords. The labels are Finglish transliterations ([Farrahi Moghaddam et al., 2010](#))

Subword image	Labels	Optimum value of (K)
	bnv	18
	n	16
	a	12
	ld	14
	lmqa	16

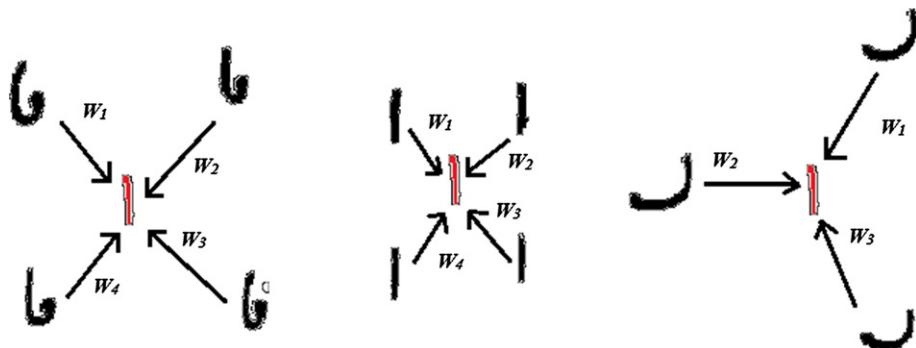


Fig. 8. Reconstruction weights of a new subword calculated in three different manifolds.

Table 5
The experimental results of the PCA+SLLE approach

	Data type	Testing set (%)
1	Binary	76.8
2	Binary+GSM	79.1

- For the new sample, select the manifold that produces the smallest amount of reconstruction error:

$$\tilde{i} = \arg \min_i e_i(x_t). \quad (24)$$

The results of this experiment are shown in Table 5. Again, it can be seen that LLE improves performance by 3%.

6. Conclusion and future prospects

With the introduction of multi-class classification techniques (Gunter and Bunke, 2003, 2004a, 2004b) and the introduction of alternative class-reduction approaches, such as equivalent binary problem technique (Farrahi Moghaddam et al., 2010), performing recognition at the letter block (subword) level is feasible. In this work, an objective approach to the recognition of letter blocks based on their image representation is introduced. Raw data from the binary images is learned using two manifold learning techniques. In order to reduce the computation complexity of the manifold learning step, the very high-dimensional input representations are first reduced using PCA. The performance of the system is then increased by adding a stroke membership map (GSM) to the binary images to represent the input of each letter block. The performance of the proposed approaches has been tested on a database from a historical Arabic manuscript with promising results. It has

been observed that including GSM in the input representation and independently applying manifold learning improves performance by 3% and 5%, respectively.

In future work, the highly multi-class nature of recognition at the letter block (subword) level will be addressed using various approaches, including the binary descriptors approach (Farrahi Moghaddam et al., 2010, 2012). In another direction, the problem of greater sensitivity to stroke variations introduced by the pixel-level representation will be addressed using nonlinear transformations on the binary images.

Acknowledgments

The authors thank the NSERC of Canada and the SSHRC of Canada (Indian Ocean World MCRI Project) for their financial support.

References

- AbdelRaouf, Ashraf, Higgins, Colin, Pridmore, Tony, Khalil, Mahmoud, 2010. Building a multi-modal Arabic corpus (MMAC). IJDAR 13, 1–18.
- Abid, Abdelaziz, 1997. “Memory of the world”: preserving our documentary heritage. Museum Int. 49 (1), 40–45.
- Barni, M., Bernaldin, J.-A., Lahanier, C., Piva, A. (Eds.), 2008. Recent advances in applications to visual cultural heritage. IEEE Signal Process. Mag. 25 (4), 1–134 (special issue).
- Antonacopoulos, Apostolos, Downton, Andy, 2007. Special issue on the analysis of historical documents. IJDAR 9 (2), 75–77.
- Ball, Gregory R., Srihari, Sargur N., Srinivasan, Harish, 2006. Segmentation-based and segmentation-free methods for spotting handwritten Arabic words. In: Proceedings IWFHR10, La Baule, France, October 23–26, pp. 20–26.
- Baudat, G., Anouar, F., 2000. Generalized discriminant analysis using a kernel approach. Neural Comput. 12, 2385–2404.
- Belongie, S., Malik, J., Puzicha, J., 2002. Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. 24 (4), 509–522.
- Bengio, Yoshua, Vincent, Pascal, 2003. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In: Thrun, Sebastian, Saul, Lawrence, Schlkopf, Bernhard (Eds.), NIPS’03, vol. 16. MIT Press, pp. 177–184.
- Breiman, Leo, 2001. Random forests. Mach. Learn. 45 (1), 5–32.
- Cheriet, Mohamed, Farrahi Moghaddam, Reza, 2012. Guide to OCR for Arabic Scripts, chapter A Robust Word Spotting System for Historical Arabic Manuscripts. Springer, ISBN 978-1-4471-4071-9.
- Farrahi Moghaddam, Reza, Cheriet, Mohamed, 2009a. Application of multi-level classifiers and clustering for automatic word-spotting in historical document images. In: ICDAR’09, Barcelona, Spain, July 26–29, pp. 511–515.
- Farrahi Moghaddam, Reza, Cheriet, Mohamed, 2009b. RSLDI: restoration of single-sided low-quality document images. Pattern Recogn. 42 (12), 3355–3364.
- Farrahi Moghaddam, Reza, Cheriet, Mohamed, 2010. A multi-scale framework for adaptive binarization of degraded document images. Pattern Recogn. 43 (6), 2186–2198.
- Farrahi Moghaddam, Reza, Cheriet, Mohamed, Adankon, Mathias M., Filonenko, Kostyantyn, Wisnovsky, Robert 2010. IBN SINA: a database for research on processing and understanding of Arabic manuscripts images. In: DAS’10, Boston, Massachusetts. ACM, pp. 11–18.
- Farrahi Moghaddam, Reza, Cheriet, Mohamed, Milo, Thomas, Wisnovsky, Robert, 2012. A prototype system for handwritten subword recognition: toward Arabic-manuscript transliteration. In: ISSPA’12, Montreal, Canada, July 3–5, pp. 1231–1237.

- Fisher, Ronald, 1936. The use of multiple measurements in taxonomic problems. *Ann. Eugenics* 7, 179–188.
- Geng, Xin, De-Chuan, Zhou, Zhi-Hua, 2005. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Trans. Syst. Man Cybern. B: Cybern.* 35 (6), 1098–1107.
- Gunter, Simon, Bunke, Horst, 2003. New boosting algorithms for classification problems with large number of classes applied to a handwritten word recognition task. In: Windeatt, Terry, Roli, Fabio (Eds.), *Lecture Notes in Computer Science: Multiple Classifier Systems (MCS'03)*, vol. 2709. Springer, Berlin, Heidelberg, pp. 161–161.
- Gunter, Simon, Bunke, Horst, 2004a. Ensembles of classifiers derived from multiple prototypes and their application to handwriting recognition. In: Roli, Fabio, Kittler, Josef, Windeatt, Terry (Eds.), *Lecture Notes in Computer Science. Structural, Syntactic, and Statistical Pattern Recognition (MCS'04)*, vol. 3077. Springer, Berlin, Heidelberg, pp. 314–323.
- Gunter, Simon, Bunke, Horst, 2004b. Evaluation of classical and novel ensemble methods for handwritten word recognition. In: Fred, Ana, Caelli, Terry, Duin, Robert, Campilho, Aurélio, de Ridder, Dick (Eds.), *Lecture Notes in Computer Science. Structural, Syntactic, and Statistical Pattern Recognition (SSPR&SPR'04)*, vol. 3138. Springer, Berlin, Heidelberg, pp. 583–591.
- Hyvärinen, A., Oja, E., 2000. Independent component analysis: algorithms and applications. *Neural Networks* 13 (4–5), 411–430.
- Kimeldorf, G., Wahba, G., 1971. Some results on Tchebycheffian spline functions. *J. Math. Anal. Appl.* 33 (1), 82–95.
- Kouropteva, Olga, Okun, Oleg, Pietikinen, Matti, 2002. Selection of the optimal parameter value for the locally linear embedding algorithm. In: First International Conference on Fuzzy Systems and Knowledge Discovery, Singapore, pp. 359–363.
- Lorigo, L.M., Govindaraju, V., 2006. Offline Arabic handwriting recognition: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (5), 712–724.
- Mahmoud, Sabri A., 1994. Arabic character recognition using Fourier descriptors and character contour encoding. *Pattern Recogn.* 27 (6), 815–824.
- Mahmoud, Sabri A., AbuHaiba, Ibrahim, Green, Roger J., 1991. Skeletonization of Arabic characters using clustering based skeletonization algorithm (CBSA). *Pattern Recogn.* 24 (5), 453–464.
- Mezghani, Neila, Mitiche, Amar, Cheriet, Mohamed, 2008. Bayes classification of online Arabic characters by Gibbs modeling of class conditional densities. *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (7), 1121–1131.
- Milo, Thomas, 2009. Arabic script: structure, geographic and regional classification. In: 33rd Internationalization and Unicode Conference (IUC), San Jose, CA, USA, October 14–16.
- Milo, Thomas, 2010. Towards Arabic Historical Script Grammar: Through Contrastive Analysis of Qur’ān Manuscripts, Writings and Writing: From Another World and Another Era (in honor of Prof. J.J. Witkam). Archetype, Cambridge, pp. 249–292.
- Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. *Philos. Mag.* 2, 559–572.
- Radhakrishna Rao, C., 1948. The utilization of multiple measurements in problems of biological classification. *J. Royal Stat. Soc. B (Methodolog.)* 10 (2), 159–203.
- Rath, Tony, Manmatha, R., 2007. Word spotting for historical documents. *IJDAR* 9 (2), 139–152.
- Roweis, Sam T., Saul, Lawrence K., 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290 (5500), 2323–2326.
- Schölkopf, B., Smola, A.J., 2002. Learning with Kernels. The MIT Press, Cambridge, MA.
- Schölkopf, Bernhard, Smola, Alex J., Müller, Klaus-Robert, 1998. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.* 10 (5), 1299–1319.
- Sobel, I., Feldman, G., 1968. A 3×3 isotropic gradient operator for image processing. Presented at a talk at the Stanford Artificial Project (unpublished but often cited).
- Steinherz, Tal, Intrator, Nathan, Rivlin, Ehud, 2000. A special skeletonization algorithm for cursive words. In: IWFHR'00, pp. 529–534.
- Valencia-Aguirre, Juliana, Álvarez Mesa, Andrés, Daza-Santacoloma, Genaro, Castellanos-Domínguez, Germán, 2009. Automatic choice of the number of nearest neighbors in locally linear embedding. In: CIARP'09. Springer-Verlag, Berlin, Heidelberg, pp. 77–84.
- Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer-Verlag, New York.
- Vapnik, V., 1998. Statistical Learning Theory. John Wiley & Sons, New York.

- Wali, Aamir, Hussain, Sarmad, 2007. Context Sensitive Shape-Substitution in Nastaliq Writing System: Analysis and Formulation, Innovations and Advanced Techniques in Computer and Information Sciences and Engineering. Springer, Netherlands, pp. 53–58.
- Wisnovsky, Robert, 2004. The Nature and Scope of Arabic Philosophical Commentary in Post-Classical (ca. 1100–1900 AD) Islamic Intellectual History: Some Preliminary Observations, Philosophy, Science and Exegesis in Greek, Arabic and Latin Commentaries, vol. 2. Institute of Classical Studies, London, pp. 149–191.
- Ye, Xiangyun, Cheriet, M., Suen, C.Y., 2001. Stroke-model-based character extraction from gray-level document images. *IEEE Trans. Image Process.* 10 (8), 1152–1161.
- Yu, Y.J., Lee, D.H., Lee, Y.B., Cho, H.G., 2003. Interactive rendering technique for realistic oriental painting. *J. WSCG* 11 (1), 538–545.
- Zhang, Shi qing, 2009. Enhanced supervised locally linear embedding. *Pattern Recogn. Lett.* 30 (13), 1208–1218.
- Zhu, Xiangbin, 2007. Shape recognition based on skeleton and support vector machines. In: Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques, vol. 2, pp. 1035–1043.

This page is intentionally left blank

Query Suggestion with Large Scale Data

Nish Parikh, Gyanit Singh, and Neel Sundaresan

eBay Research Labs, San Jose, CA, USA

Abstract

Explosive growth of information has created a challenge for search engines in various domains to handle large scale data. It is still difficult for search engines to fully understand user intent in many scenarios. To address this, most search engines provide assistive features to the user which help the users in managing their information need. Query Suggestion (Related Searches) is one such feature which is an integral part of all search engines. It helps steer users toward queries which are more likely to help them succeed in their search missions. There has been extensive research in this field. In this chapter, we discuss state-of-the-art techniques to build a Query Suggestion system. Specifically, we describe the strengths and limitations of different approaches. We also describe salient characteristics of large scale data sets like query corpora and click-stream logs. We walk the reader through the design, implementation, and evaluation of large scale Query Suggestion systems in practice. We show how challenges related to sparsity in the long tail, biases in user data, and speed of algorithms can be tackled at industry scale.

Keywords: Query Suggestions, Related searches, Query reformulation, Search, Big data, Recommender system

1. Introduction

In today's world after the explosion of the World Wide Web, search engines are ubiquitous. Be it the WWW, product sites, social network sites or intranets search engine is a primary window for the users into these sites. Search plays an important role not only in discovering content, information, audios, videos, images but also members, products, deals, and navigational links based on the site the user is interacting with.

Typically, users express their need or intent to search engines in the form of searches or queries. The search engines in turn try to map the intent to matching documents, products, or other information content that can be of utility to the user.

For most purposes in this chapter we will discuss text queries, however user queries in today's world could be in the form of text, images, or voice.

The explosive growth of web information has not only created a crucial challenge for search engine companies to handle large scale data, but also increased the difficulty for a user to manage her information need. While more and more sophisticated algorithms have been applied by ever evolving search engines to understand the user intent better and better and to enrich user search experience, it is still admittedly difficult for search engines to fully understand users' search intent in many scenarios. Most search engines are still inherently Boolean (Frants et al., 1999) in nature, try to match user entered text to publisher text and do not do much semantic interpretation. Also, it has become increasingly difficult for a user to compose succinct and precise queries to represent her needs. It is not always easy for users to formulate effective queries to search engines. As formulating the best queries for a given task is a burdensome task for users and most users are not proficient at it, it is a common practice for search engines to provide features which can assist the users in this task.

Some such features are **Query Auto-Complete**, **Query Rewrites**, and **Related Searches** (search short-cuts). Search engines employ query rewrites to automatically rewrite user queries to find more suitable documents matching the users need. This could include automatically searching for synonyms of terms used by users or automatically looking for morphological variants of terms used by users. For example, if a user searches for *photography* the engine might decide to include results for *photograph* and similarly if the user searches for *ps2* the engine might want to automatically include results for *playstation 2*. Sometimes engines also use substitution. For e.g., a users query for *britannie spars* which may not return any matching results might automatically be substituted by the query *britney spears*.

Auto-Complete systems typically show query completion options in drop-down menus as the users start typing their queries or partially specify their queries. This helps steer the users toward queries which are more likely to satisfy their intent and helps save typing time or number of keystrokes or clicks the users have to undergo to reach their goal. Various signals are used by Auto-Complete systems to show queries to users, but most of these systems work on prefix matching and show popular query completions based on what the user might have already typed in. An example of such a system in action is shown in Fig. 1. **Related Search** recommendations is another mechanism where users are shown related queries which might help them in finding relevant results for their need or results related to their interest. All these features are important to help search engine users in their information seeking activities or search missions. Sometimes these feature names viz. **Query Rewrites**, **Query Auto-Complete**, **Related Searches** and also some other features like **Advanced Query Builders** (which let users build complex queries based on Boolean criteria—inclusions, exclusions, and wildcards) are used synonymously and commonly referred to under the umbrella of **Query Suggestions**. However, for the purpose of this chapter we will discuss **Related Searches Recommendations** only and will refer to those as **Query Suggestions**.

Query Suggestion is an interactive approach for search engines to better understand users' needs. There are always differences between searcher and publisher vocabulary (Singh et al., 2011b) as well as user intent and a search engine's

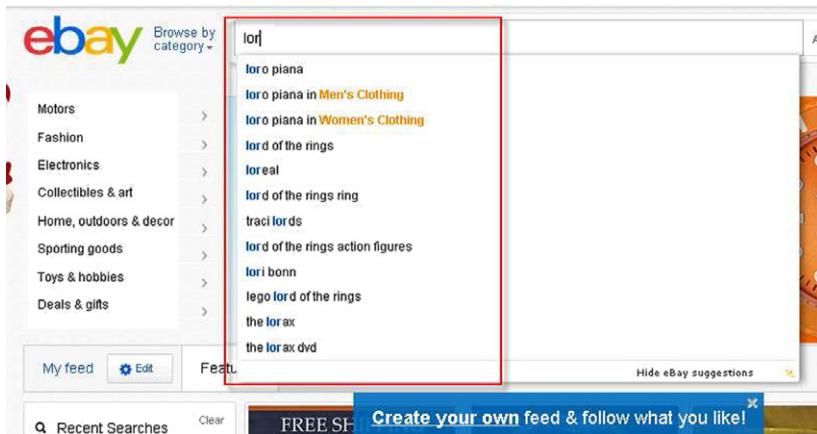


Fig. 1. Auto-complete Query Suggestions shown in a drop-down menu when a user visits the eBay.com home page and types in *lor* in the search box.

understanding of the intent. For e.g., a publisher might list a product as *universal travel adapter* whereas a product searcher might search for *travel adapter france* and a purely Boolean search engine may not be able to match this searcher and publisher. Also for a query like *java* the user's intent might be to look for coffee but the search engine might interpret the intent as being related to the Java programming language. Query Suggestion modules help bridge such gaps. They enable the searchers to have a dialog with the search engine to be able to reach their final goal efficiently.

Query Suggestion is a core task for large industrial search engines. As a result Query Suggestion module is an integral part of every search engine and manifests to users as an important tool in improving their search experience and finding relevant results to their queries. Query Suggestion is observed on web search engines (Fig. 2), e-commerce search engines (Fig. 3), social content sites (Fig. 4) as well as in enterprise search and some versions of e-mail and desktop search. Typically, the search result pages for a query would contain a list of suggestion queries which might help satisfy the visitor's information need more appropriately.

In general, Query Suggestions is an important assistive features in all domains. It can serve multiple purposes and help the users to focus on their intent in a fine-grained fashion or explore new content. For focused visitors it can help them short-circuit their path to success i.e., help them reformulate their queries appropriately so that the desired object (item, web page or other) can be retrieved in a shorter time with lesser effort. For exploratory visitors it provides a means of discovery (Al Hasan et al., 2011). The main task of Query Suggestions is to improve the overall search engagement for the users. Broadly the utility of Query Suggestions can be classified into six classes described below.

- *Specialization* (Narrowing Down on Intent): This provides an option to the users to refine their searches. For e.g., if the user enters a broad query like *nikon* and gets inundated with information the engine might suggest some options to the user to narrow down the intent. For e.g., *nikon camera*.

WEB IMAGES VIDEOS MAPS NEWS MORE



twinkies



4,530,000 RESULTS

News about twinkies

bing.com/news



[Oakland joins nation in mourning Twinkie death after Hostess bankruptcy](#)

San Francisco Gate · 7 hours ago

Within twenty minutes, it was gone. When word spread that there was half a box of **Twinkies** still on a shelf at the A&A Market on Sunday afternoon, people of...

See also: [More stories](#) · [Top stories](#) · [Related blogs](#)

RELATED SEARCHES

[Deep Fried Twinkie](#)
[Chocolate Covered Twinkies](#)
[Twinkie Shelf Life](#)
[Twinkie Desserts](#)
[Twinkie Wedding Cake](#)
[Twinkie Diet](#)
[Twinkie Man](#)
[Twinkies Nutrition Information](#)

Twinkies

www.twinkies.org ▾

It is never a good time to have a dispute with a big company and there is little doubt that this will not be easy. But sometimes one must act on an inner ...

Fig. 2. Query Suggestions shown on bing.com for the search term *twinkies*. Bing.com shows this feature on the right of the page. Around eight suggestions are shown in a vertical rank order. Also, reverse highlighting is employed. i.e., terms in suggestion queries which are not present in original query are highlighted.

ebay

khloe kardashian All Categories Search Advanced

Related Searches: **khloe and lamar**, **dash clothing**, **kim kardashian**, **christian louboutin shoes**, **kardashian**, **khloe kardashian store**, **kourtney kardashian**

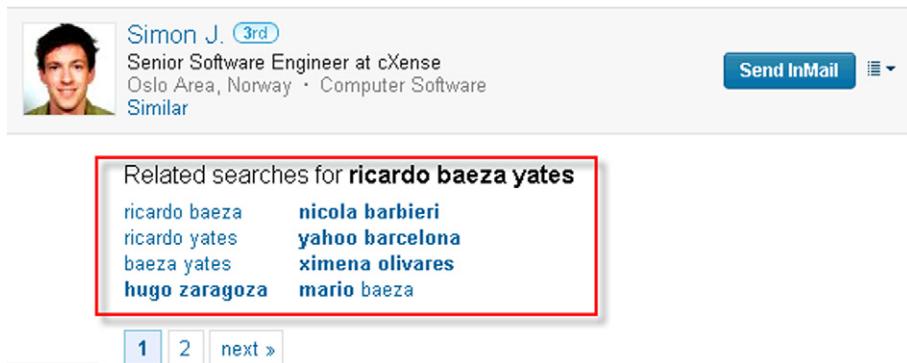
▼ Categories 449 results found for khloe kardashian [Save search](#)

Books (212)	All items	Auctions only	Buy It Now	Products & reviews Beta	Customize view
Magazine Back Issues (17)					
Nonfiction (29)	View as:				
Fiction & Literature (12)					Sort by: Best Match
More					Page 1 of 9
Clothing, Shoes & Accessories					
Women's Clothing (28)					
Women's Shoes (9)					
More					


4.5CT Khloe Kardashian Inspired Wedding Ring Set CZ Sterling Silver Emerald-Cut
\$139.00

Returns: Accepted within 14 days
[More Options](#)

Fig. 3. Query Suggestions shown on ebay.com for the query *khloe kardashian*. Suggestions are shown at the top of the page below the search box from left to right in a ranked order. Seven suggestions are shown.



The screenshot shows a LinkedIn search results page for the query "ricardo baeza yates". At the top, there is a profile card for "Simon J. 3rd" with the title "Senior Software Engineer at cxSense" and location "Oslo Area, Norway · Computer Software". Below the profile, there is a section titled "Related searches for ricardo baeza yates" containing the following suggestions:

- ricardo baeza
- nicola barbieri
- ricardo yates
- yahoo barcelona
- baeza yates
- ximena olivares
- hugo zaragoza**
- mario baeza

Below the suggestions, there are navigation links: "1" (selected), "2", and "next »". A red box highlights the "Related searches" section.

Fig. 4. Query Suggestions shown on linkedIn.com for the query *ricardo baeza yates*. Suggestions are shown at the bottom of the page. Eight suggestions are shown. Reverse-highlighting is used. Some search engines like linkedIn.com choose to show Query Suggestions at the bottom of the page. By the time members have scrolled to the bottom of a search results page, it is reasonable to assume that they have not found what they are looking for at least on the first page. This is when Query Suggestions could be most useful (Reda et al., 2012).

- *Generalization*: If a user searches for *ipod nano 4gba* suggestion like *mp3 player* which generalizes the user intent to some extent can give the user a chance to look at more content.
- *Vocabulary Match*: For engines that do not use extremely sophisticated query rewrite a query such as *garnet* will not match content described as *january birthstone*. A product listed as *martin guitar* will not match the query *acoustic guitar* although *martin guitar* is a specialization of *acoustic guitar*. Also a product listed as *oak table* will not match the query *wood table*. Query Suggestions can help close this gap. As described by Baeza-Yates and Tiberi (2007) Users searching for the same information may phrase their queries differently. Often, users try different queries until they are satisfied with the results. In order to formulate effective queries, users may need to be familiar with specific terminology in a knowledge domain. This is not always the case: users may have little knowledge about the information they are searching, and worst, they could not even be certain about what to search for. As an example, a tourist seeking for summer rental ads in Chile may not know that the vast majority of such ads in the Web are for apartments in Vina del Mar, a popular beach in the central part of Chile. In contrast, local users may have the expertise to submit queries with the term *vina del mar*, when they are looking for a location to spend their vacations. Query Suggestions can help novice users learn about expert vocabulary.
- *Exploration (Parallel Moves)* (Boldi et al., 2009): Search systems use Query Suggestions to let the users explore or see content which they may not be aware about. For e.g., in product search a Query Suggestion of *eucalyptus wallflower* might be shown in response to a users query for *pear wallflower* for a user looking for bath and body works health and beauty products. In this case the user may not have known that such a fragrance existed but might be happy to explore it.

A user searching for *nikon* may also be suggested *canon* as an alternate brand to explore.

- *Disambiguation:* Search engines allow users to specify queries simply as lists of keywords following the approach of traditional information retrieval systems (Baeza-yates et al., 2004; Ricardo, 1999). However, it is not always easy for users to formulate effective queries to search engines. One reason for this is the ambiguity (Cronen-Townsend and Bruce Croft, 2002; Song et al., 2007) that might arise in many terms of a language. Queries having ambiguous terms may retrieve documents which are not what the users intended to retrieve. On the other hand, users typically submit very short queries (Singh et al., 2011b; Silverstein et al., 1998) to search engines and short queries are more likely to be ambiguous. Query Suggestions can provide options to the user which can help in disambiguation. For e.g., in response to a user query *usc* Query Suggestions like *university of southern california* and *university of south carolina* can help disambiguate user intent.
- *Error Correction:* Some search engines employ Query Suggestions to suggest spell corrections or phonetic corrections in response to user queries. However, many modern search engines seem to tackle Spell Correction separately with explicit messaging and presentation.

Query Suggestions are an excellent assistive tool for users. All sites strive to design Query Suggestion systems which can delight the users and improve engagement on the website. There are a multitude of challenges involved in the design of such systems. Some of the challenges are mentioned below.

- *Presentation and User Interface:* Today's search engines try to provide as friendly an interface as possible. They have to make the right choices for interfaces from a usability perspective. The interface must be understandable by and appealing to a wide variety of people of all ages, cultures and backgrounds and for an enormous variety of information needs as described by Hearst (2009). Every choice made in this regard is important. Messaging for the feature is important. In practice we see engines messaging it as **Related Searches** or **People Also Searched For** or **Something Different**. We also observe that some engines show suggestions at the top of the page, some at the bottom, some at the right and some even at more than one location on the page. Another design choice is the amount of real-estate on the page that should be devoted to the feature (which decides how many suggestions can be shown), how the terms in the suggestions should be highlighted (see 2). Some systems present Query Suggestions as a flat list whereas some propose showing it in a clustered or structured way. One such example of structured representation of Query Suggestions is the SparQS System (Kato et al., 2012). We describe later in this chapter (see Table 3) how a simple change in term highlighting can impact the interaction of users with the feature.
- *Metrics:* Websites need to decide on what metrics they would like to optimize on by providing features such as Query Suggestions. For example, product sites could offer exploratory Query Suggestions but it might distract the user from the real task at hand for e.g., buying the product that she first looked for, instead of exploring alternative products and this could be detrimental in some ways for the product site. E.g., for a user searching for *campfire popcorn* the suggestion *gps* might be

interesting as both of those are related to the camping theme. However, a user pursuing to click on that suggestion might get distracted from buying *campfire popcorn*. Some common evaluation methodologies and metrics used for Query Suggestion systems are described in Section 4.

- *Data and Algorithms:* Query Suggestion systems can be built using various signals and filters that capture several dimensions of relatedness from user searching activities or other corpuses of data. The basics of all such systems are usually to find queries that are similar to the user entered query in many dimensions. They might differ in some dimensions which would achieve one of the goals described above viz. correction, specialization, generalization, disambiguation, exploration, or bridging the vocabulary gap. Suggestion queries are typically related to original query in a semantic way. It has been shown that a good query suggestion system should consider a handful of features, but in most cases it is important to ensure that the semantics of the suggested query do not drift too much from the original one (Mei et al., 2008). Suggestions are typically queries similar to the original query and many a times obtained by analyzing query logs (either clustering queries or looking at frequent re-phasings). However, there has also been work which shows how Query Suggestions can be generated in the absence of query logs (Bhatia et al., 2011). Various approaches used to generate Query Suggestion recommendations are described in Section 3. There are always considerations around freshness of data (Baraglia et al., 2009) and seasonal effects (Parikh and Sundaresan, 2009) on recommendations. There is also an emphasis on optimizing the Query Suggestion sets to improve the satisfaction of the average user. This entails diversifying the suggestions as described in Sydow et al. (2011), Ma et al. (2010), and Zhu et al. (2010)).
- *Personalization:* This is an active area of research. By using user-based signals in addition to query signals more appealing Query Suggestions can be shown to the users. This is especially important for domains like e-mail search and desktop search (Bhatia et al., 2011). Some published research work in this area can be found in (Mei et al., 2008; Cao et al., 2008; Chatzopoulou et al., 2011).

The chapter is laid out as follows. In Section 2 we describe the terminology that is used throughout this chapter. Section 3 describes different methods that can be used to generate Query Suggestions. Strengths and limitations of the different approaches are also described. Section 4 illustrates typical metrics used by search engines to evaluate Query Suggestion systems. In Section 5 we show the nature of large scale data and the challenges involved in working with the same. In Section 6 we discuss in detail the design and implementation of a practical Query Suggestion system. Various challenges and how they can be tackled at industry scale is described. Finally, we summarize in Section 7.

2. Terminology

In this section we define and describe the terms which will be used for the rest of the chapter. We will use items, url, and docs interchangeably to describe the entities on which search is being performed.

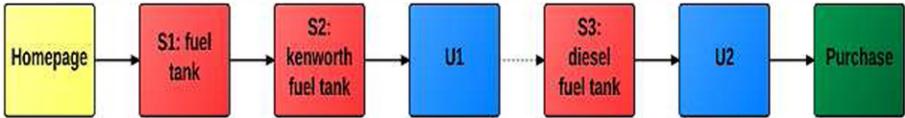


Fig. 5. Example session from click-stream logs. Search block are tagged using following convention name:query. For example S1 event was search done using query “fuel tank.” Time increases from left to right. U1 and U2 are clicked URLs. Solid arrow represent browsing via a user click.

Query is defined as user-typed terms which are used as input to look up matching items in item corpora. For e.g., “*khloe kardashian*,” “*graduation wine bottle opener*.” We will refer to a query (of k length) as $q = (t_1, \dots, t_k)$, where t_i is the i th *term* or *word* in the query. *Constraints* are other configurable parameters allowed by the search engine to be fixed by the user. We will refer to them as \mathbb{C} . Constraint type may vary from domain to domain. For e.g., in image search, image-size, image-quality are few of the supported constraints. In e-commerce search price, shipping-cost, location are few of the supported constraints. In web search, file extension, time is few of the supported constraints.

Search result page (SRP) is defined the page generated by the search engine using query and constraints. The input to generate a SRP is the pair or query term and constraints i.e., (q, \mathbb{C}) , where \mathbb{C} is the tuple containing value for all the constraint values set by the user. We will use q and (q, \mathbb{C}) interchangeably. *Recall set* of the query is defined as set of URLs matching to the query. We will denote recall set of the query q as R_q . *Impressed URLs* are the list of all URLs that appear on the page as result to the query. This number is normally configured to be 25 or 50 for search engines. We will call this set of url for query q as I_q . The *Clicked URLs* are the set of URLs that the user clicked on from the impressed URLs. For a query q we will refer to the clicked URL set as C_q (see Fig. 5).

A *session* is defined as all the user activity on the site such that the time difference between any two action is less than 30 min. Logs of these activity are also called *click-stream* logs or *session-logs*. More precisely, a page visited by user u is referred as $\langle p, u, t \rangle$, where p is the page visited and t is the time-stamp. A session of length n is defined as ordered list of such tuples $(\langle p_1, u_1, t_1 \rangle, \dots, \langle p_n, u_n, t_n \rangle)$ s.t. $t_i - t_{i-1} < 30$ min $\forall i$. Figure 5 contains an example session from the session-logs.

Search log or query log records the queries and the actions of the users of search engines. It comprises of set of tuple of type $\langle u, s_u, q, t, I_q, C_q \rangle$, where u is the user, s_u is the session id, q is the query-term and constraint pair, t is the timestamp, I_q is the set of URLs shown to the user, and C_q is the set of URLs clicked by the user.

Search Trails: Search trails in our logs are analogous to search trails described by White and Huang (2010). Search trails begin with a search being issued by the user and terminate at a point where user has finished her searching activity.

By *platform*: we refer to the device or OS used by the user to access the search engine. Web-browser on PCs, mobile devices, and tablets are most of the common platform to access the search engine.

3. Approaches to generation of Query Suggestions

In this section we describe many of the similarity scoring techniques used to generate query suggestion. We have divided these techniques by the data they use. Those similarity scoring techniques which only use query text are described in textual similarity subsection. Similarity scoring function which make use of the matched url set of the query in some way is described in content-based scoring subsection. If the scoring function requires query-click data or graph they are described in query-log based scoring schemes. In the end if scoring function makes use of sessions and search's position in the session they are described in session-based scoring subsection.

Some paper for e.g., [Mei et al. \(2008\)](#) and [Craswell and Szummer \(2007\)](#) use the similarity score described below as ranking scheme to generate query suggestion. But other papers ([Baeza-yates et al. \(2004, 2000\)](#); [Baeza-Yates and Tiberi, 2007](#)) use the similarity score described below to induce a clustering. For example [Baeza-yates et al. \(2004\)](#) uses k -means iteratively to generate query cluster using the similarity measure. Whereas [Beeferman and Berger \(2000\)](#) uses iterative agglomerative clustering using the similarity measure. Once these clusters are generated the Query Suggestions are generated from within the cluster.

Where as papers like [Al Hasan et al. \(2011\)](#), [Parikh and Sundaresan \(2008\)](#) and [Reda et al. \(2012\)](#) use hybrid approach where they use more than one similarity measure described below (textual, session-based, and query-log based). Then the query suggestion is generated using the combination of all those similarity scores.

3.1. Textual similarity among queries

Features based on the terms present in queries are the simplest features to use for Query Suggestions. Various algorithms have been proposed and published that look at terms present in queries to find similarity among queries based on such features and then use similar queries as Query Suggestions.

Most such approaches look at terms in queries and find some distance metric to measure dissimilarity or similarity between them. Some of the popular approaches are described below.

- *Levenshtein Distance Based on Terms in Queries:* Because search engine users often reformulate their input queries by adding, deleting, or changing some words of the original query string, Levenshtein Distance ([Gilleland et al., 2009](#)) which is a special type of edit distance can be used to measure the degree of similarity between query strings. It defines a set of edit operations such as insertion or deletion of a term, together with a cost for each operation. The distance between two queries is defined to be the sum of the costs in the cheapest chain of edit operations transforming one query into the other. For example, the Levenshtein distance between *tolkien hobbit* and *tolkien hobbit 1st edition* is 2. Hence, the similarity of two queries can be measured based on the Levenshtein Distance between them and was defined by [Shi and Yang \(2006\)](#) as:

$$\text{Similarity}_{\text{Levenshtein}}(q_1, q_2) = 1 - \frac{\text{Levenshtein_Distance}(q_1, q_2)}{\max(\text{wn}(q_1), \text{wn}(q_2))}, \quad (1)$$

where $\text{wn}(\cdot)$ is the number of terms(or characters for Chinese queries) in a query.

- *Exponential Distance based on Terms in Queries:* Such a distance function is described by [Parikh and Sundaresan \(2008\)](#) for e-commerce queries. In [Parikh and Sundaresan \(2008\)](#), a simple similarity measure is used to relate queries based on textual similarity. A graph called Term Connection Graph is created where each query is a node and weighted edges exist based on the proposed distance function. Every query is represented as a set comprising of terms found in the query. So, a query Q would be represented as $Q = W_q = W_1, W_2, \dots, W_n$, where $W_i, i = 1, 2, 3, \dots, n$ are the unique terms in the queries and n is the total number of unique terms in query Q . For every query Q , all queries Q_c such that $W_q \subset W_{qc}$ are found. These are all the queries that can be formed by adding new terms to query Q . Q is connected to every Q_c in the Term Connection Graph with an edge. Also all queries Q_l such that $W_{ql} \subset W_q$ are found. These are all queries that can be formed by dropping terms from query Q . All such queries Q_l are connected with the node for query Q in the Term Connection Graph. Queries formed by adding new terms to the original query are usually specializations of the original query, whereas queries formed by dropping terms from the original query usually tend to be generalizations of the original query. One bit of meta-data on every edge between any two pairs of nodes Q_a and Q_b in the Term Connection Graph is also introduced. This conveys whether the relationship from Q_a to Q_b is a generalization or specialization. As the edges are bidirectional in the graph, traversal in one direction would give a generalization whereas traversal in the other direction would give a specialization. How such a graph can be efficiently built using an inverted index of queries is also described in [Parikh and Sundaresan \(2008\)](#). The dissimilarity between queries increases as the number of differing terms in the queries increase and the proposed distance metric tries to capture the same. The term distance between two queries D is the number of terms by which the two queries differ. Then an exponential scoring mechanism is used to normalize the similarity score between 0 and 1. The similarity between two queries Q_a and Q_b based on the Term Connection Graph is defined as:

$$T_s = \frac{1}{2^D}, \quad (2)$$

if $W_{qa} \subset W_{qb}$ or $W_{qb} \subset W_{qa}$, otherwise the similarity is zero and the corresponding edge between Q_a and Q_b will not exist in the Term Connection Graph.

For a corpus of 17 e-commerce million queries mapping to 17 million nodes in the Term Connection Graph in [Parikh and Sundaresan \(2008\)](#)) it is shown that there are only 500,000 nodes which do not have any connections. Those are the rare queries or misspelled queries or some non-sensical robot generated queries. Otherwise, the degree distribution for the nodes follows a power-law distribution. The Term Connection Graph is not very dense and has around 168 million edges.

- *Distance based on statistical importance of terms in queries:* Previous approaches described do not differentiate between different kinds of terms in queries. An approach which weights the terms in queries is described in [Reda et al. \(2012\)](#). Reda et al. describe Query Suggestions based on the term overlaps in queries and discuss the adjustments that can be made to obtain better recommendations. As shown above with millions of unique queries, term overlaps are rather common and the difficulty is in identifying a set of queries that are meaningfully overlapping. To

tackle the problem, the authors start out by grouping unique queries together and counting their occurrence which results in pairs of form (query, count) which are used as features for further processing. Also they do not use simple white space tokenization for queries. For each unique query, they identify the terms that make up the query, but the tokenization process incorporates some optimizations. Firstly, stop words from the set are removed. The assumption here is that stop words do not add much to the meaning of the query. They also remove very short terms. Although this can lead to some meaningful terms being removed they observe overall improvement in quality of recommendations when such feature cleaning is performed. Using the query pairs and their corresponding terms, a new set of tuples of the form (term, query, count) are generated and then grouped based on terms as the key. This gives an indication of the number of queries in the query corpus which contain a given term. Not all terms in queries are equally important (Jones and Fain, 2003). For example, for the query *ipad mini*, the term *ipad* is more important to the query than the term *mini*. Accordingly it is more useful to make recommendations that look like *ipad 2* rather than *mini skirt*, although both queries share a term with the original query. To measure the importance of a term t in the query, Reda et al. (2012) use a flavor of an inverse document frequency as follows:

$$\text{IDF}(t) = \log \frac{M - Q(t) + 0.5}{Q(t) + 0.5}, \quad (3)$$

where M is the total number of unique queries and $Q(t)$ is the total number of queries that contain the term t . The more common a term is across different queries, the less significant it is considered for any one of them.

Queries are short hence understanding semantic meaning is difficult. Hence, such term based on features and algorithms are seldom used in practice in isolation. Using purely text-based features, one can never find a suggestion *keira knightley* for *jessica alba* or *zune* for *ipod* or *zune* for *mp3 player* as although these pairs of queries might be related at some semantic level and might be helpful to the user for exploration, they do not share any term in common. On the other hand, such algorithms will find relationships like *apple player* for *apple dishes* as they share the term *apple*, although there may not be any overlap in intent or semantics of those two queries. The next sections describe how short queries can be mapped to concepts or expanded context. This enables defining richer similarity functions thereby improving the quality of Query Suggestions.

3.2. Content-based similarity

Standard text similarity measures perform poorly on query similarity as queries are small piece of text. This leads to data sparseness and lack of context. Sahami and Heilman (2006) and Metzler et al. (2007) discuss strategies to measure similarity between short text snippets even when those snippets don't have any word in common.

As defined in Section 2 for a query q , R_q is the set of URLs matching the query. This set is then ranked by relevance to q and top m URLs $\{d_1, \dots, d_m\}$ are picked;

where m is a prefixed constant, normally chosen to 100. TF-IDF term vectors v_i are computed for each document $d \in \{d_1, \dots, d_m\}$. Then q is projected up to a term vector $\hat{q} = (q_1, \dots, q_n)$ where $n = \text{Number of unique terms}$. \hat{q} is computed as centroid of all vectors v_i computed from documents matching for query q . Content-based similarity between q_1, q_2 is computed as L_2 norm of vectors \hat{q}_1, \hat{q}_2 . Metzler et al. (2007) provides a detailed study of different choices of similarity score function using these vectors.

3.3. Query-log based similarity

In this subsection we describe various similarity scores built using query-log. A query log containing set of n tuple of type $\langle u_i, s_i, q_i, t_i, I_{q_i,i}, C_{q_i,i} \rangle$, where u_i is the user, s_i is the session id, q_i is the query, t_i is the time-stamp, $I_{q_i,i}$ is the set of URLs shown to the user, and $C_{q_i,i}$ is the set of URLs clicked by the user. For a query q occurring at-least once in the query log, we define aggregate click URL set (AC_q) as all the URLs that were clicked atleast by one user on SRP generated by query q .

$$AC_q = \{url \in \text{URLs} \mid \exists i \text{ s.t. } q_i = q \text{ and } url \in C_{q_i,i}\} = \bigcup_{\forall i \text{ s.t. } q_i = q} C_{q_i,i}. \quad (4)$$

A query-clicked url bipartite graph is induced from the aggregate clicked URLs. Nodes in one part of the graph constitutes of all the queries from query log. Clicked URLs comprise of nodes on the other side. An edge is drawn from node q to node url if \exists user u who clicked on url from q . Weighted version of these graphs is also made in some cases. Weight of the edge (q, url) is number of clicks url got in query-log from q . Figure 6 shows a small sample of such graph.

3.3.1. Query-clicked URL set based similarity

Query-clicked URL set is same as neighborhood of the query node in the above mentioned graph. For two queries q_1, q_2 in the graph lets denote their neighborhood

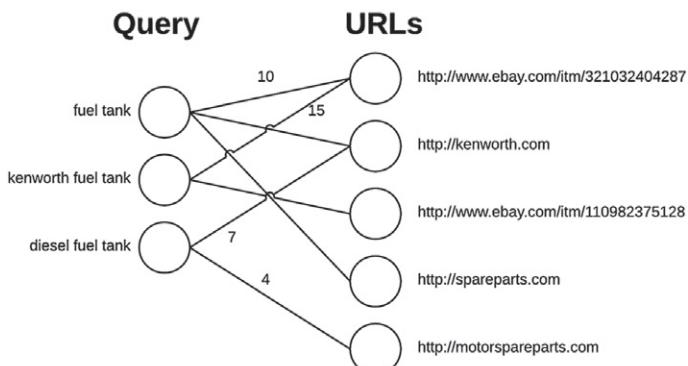


Fig. 6. Query-clicked-URL bipartite graph. Left-hand side are the queries and right-hand side are the urls. Edge $e = (q, u)$ denote that url u got a click from the query q . Weight of the edges (q, u) ($w_{q,u}$) denotes number of clicks.

by $N(q_1), N(q_2)$. One thing to note is that the neighborhood of the query q is same as aggregate click set AC_q .

Similarity is inferred from the overlap of AC_{q_1}, AC_{q_2} . Beeferman and Berger (2000) use such similarity measure. More formally

$$\text{similarity}(q_1, q_2) = \begin{cases} \frac{|AC_{q_1} \cap AC_{q_2}|}{|AC_{q_1} \cup AC_{q_2}|}, & \text{if } |AC_{q_1} \cup AC_{q_2}| > 0, \\ 0, & \text{otherwise.} \end{cases}$$

Another strategy used is to use this neighborhood to project query q up in a much larger space to \hat{q} . For example Baeza-Yates and Tiberi (2007) use vector representation of the neighborhood, where dimension of the vector is number of URLs. Similarity between two queries is computed by taking cosine of the corresponding vector.

Another way to consume URL neighborhood set is induce large term vectors for query using the aggregate click set. Given a query q , and a URL u , let $f_{q,u}$ be the fraction of clicks in the answers of q . Let $F_{t,u}$ be the number of occurrences of term t in URL u . We define a vector representation for q as $\hat{q} = (q_1, \dots, q_k)$, where k is the number of unique terms.

$$q_i = \sum_{u \in \text{URL}} \frac{f_{q,u} \cdot F_{t_i,u}}{\max_t F_{t,u}},$$

where the sum ranges over all clicked URLs. Note that above representation changes the inverse document frequency by click popularity in the classical *tf-idf* weighting scheme.

Different notion of vector similarity, for example cosine or Pearson correlation, can be applied over the above vector representation. Such similarity score is used in Reference (Baeza-yates et al., 2004).

3.3.2. Shortest-path and distance as similarity

Another graph-based similarity measure used is shortest path length or graph distance. The unweighted distance of two different query node is seen as distance between two queries. Smaller the graph distance more related the queries are considered. The graph distance is computed as the length of the shortest path from node q_1 to q_2 .

3.3.3. Random walks

Craswell and Szummer (2007) defines a lazy random walk over the bipartite graph. The self-transition probability s is the probability that user is going to stay at the same query in the next step of the walk. Self-transitions allow the random walk to stay in place, and reinforce the importance of the starting point by slowing diffusion to other nodes. The transition probability from node i to j denoted by $P_{t_{i+1}|t_i}(j|i)$ is defined as ratio of weight of edge (i,j) with the out degree of node i discounted by the lazy factor.

$$P_{t_{i+1}|t_i}(j|i) = \begin{cases} (1-s) * \frac{w_{i,j}}{\sum_k w_{i,k}} & \forall i \neq j, \\ s & \text{when } i = j. \end{cases}$$

Given a query q scoring of other queries is done by simulating a backward random walk as described above for x number of steps. Authors of [Craswell and Szummer \(2007\)](#) report that they achieved best results for 11-step process.

3.3.4. Hitting time

[Mei et al. \(2008\)](#) use a slightly different variation of random walk over the same graph. They induce a random walk over the query node subgraph. For two query nodes i, j they define the transition probability from i to j denoted by $P_{t_{i+1}|t_i}(j|i)$ as sum of probability over all two edge path from i to j . Each path probability is defined as product of probability of each edge. More precisely it can be written as:

$$P_{t_{i+1}|t_i}(j|i) = \sum_{k \in \text{URL}} \frac{w_{i,k}}{\sum_l w_{i,l}} \cdot \frac{w_{k,j}}{\sum_l w_{k,l}}.$$

For the graph $G = (V, E)$, let A be a subset of V . Let X_t denote the position of the random walk at discrete time t . The hitting time T^A is the first time that the random walk is at a vertex in A .

$$T^A = \min\{t : X_t \in A, t \geq 0\}.$$

The mean hitting time h_i^A is the expectation of T^A under the condition $X_0 = i$, that is,

$$h_i^A = E[T^A | X_0 = i].$$

In other words expectation of hitting times of nodes in A given that the starting point of random walk is i . h_t^q for some prefixed number of step t is computed and is used as similarity to query q .

3.4. Session-based similarity

In this subsection we describe other bucket of behavioral features. These features not only use the clicks on the query but also use the whole session information. Figure 7 shows three example sessions. These features can be roughly bucketed into three buckets. For some pair of queries which occur in same session, features are computed with respect to various actions. An action may be a click or purchase or transaction or just occurrence (raw frequency of pair). As these feature are collected per query pair, they are interpreted as similarity scores as well.

3.4.1. Query adjacency

Query adjacency score is described in [Parikh and Sundaresan \(2008\)](#). Session logs are used to infer score for pair of queries. Whenever the pair (q_i, q_j) occur in consecutive order in a session, scores are updated based on outcome of q_j . For each *action* taken by the user on q_j a score is added for q_i, q_j, action triplet; where action taken by

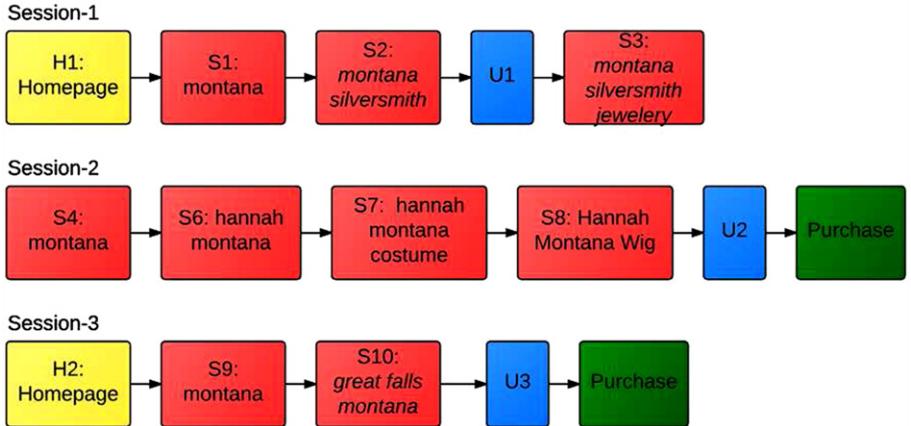


Fig. 7. Shows three example sessions from the session logs. Each session contains the query *montana* in it.

user $\in \{occurrence, click, success\}$. We will represent them in short hand by $s(q_i, q_j, a)$, where $a = \{\text{freq}, \text{click}, \text{success}\}$. For each occurrence of $q_i \rightarrow q_j \rightarrow action$ triplet in the Session 1 is added to $s(q_i, q_j, a)$ its score.

3.4.2. Query short-cutting

Query short-cutting is an extension of query adjacency. This scoring scheme is designed to take into account for queries which are issued with some gap within each other. This score is parameterized by the fixed parameter α also known as discount factor. The discount factor is introduced to counter-balance loss of context as the pages between two searches increases. For two searches q_1, q_2 with d distance apart in the session; some function of the discount factor α and distance between the two searches (d) is used to update the similarity score for q_1, q_2 . Zhang and Nasraoui (2006) uses an exponential discount α^d where $\alpha < 1$.

$$\text{discount}(d, \alpha) = \begin{cases} 0, & \forall d \geq 3, \\ \alpha^d, & \text{when } d \leq 2. \end{cases}$$

A1 Hasan et al. (2011) uses harmonic discount with $\alpha = 1$.

$$\text{discount}(d, \alpha) = \begin{cases} 0, & \forall d > 8, \\ \frac{1}{d}, & \text{when } d \leq 8. \end{cases}$$

For each occurrence of $q_i \rightarrow_1 q_{i+1} \rightarrow_2 \dots \rightarrow_d q_j \rightarrow action$ the similarity score $s(q_i, q_j, a)$ is incremented by $\text{discount}(d, \alpha)$. Various session based similarity scores computed for the three sessions described in Fig. 7 are shown in Table 1.

Table 1

Shows session-based similarity scores for few pair of queries (left query, right query) using different scoring mechanism for the three sessions shown in Fig. 7

Left query	Right query	Action	Adjacent	Short-cut	
				Expo	Harmonic
montana	montana silversmith	freq	1	α	1
montana	montana silversmith jewelery	freq	1	α	1
montana	montana silversmith jewelery	click	0	α^2	1/2
montana silversmith	montana silversmith jewelery	freq	1	α	1
montana	hannah montana	freq	1	α	1
montana	hannah montana	freq	1	α	1
montana	hannah montana costume	freq	0	α^2	1/2
montana	hannah montana wig	freq	0	0	1/3

4. Evaluation methods of QS

In this section we describe the standard techniques of evaluating a query suggestion algorithm. We also describe few key measures on which different algorithm's performance is compared.

Query Suggestion is evaluated in an (a) **offline** and (b) **online** settings. In an *offline* setting; for set of pre-chosen (random) queries, suggestions are computed. Multiple human judges are asked to score the recommendations based on relevance. In an *online* setting, query suggestion is plugged into search engine used by large number of users. The behavior of users are monitored over time. In this setting there is also atleast one control bucket of users without the algorithm experience to provide baselines and noise levels for measures that are being computed. In both offline and online setting, coverage and impression remain key measures.

4.1. Coverage and impression

Coverage reflects the ability of query suggestion algorithm to generate recommendation. It is computed as the ratio of queries for which suggestions are generated by total number of queries for which algorithm was invoked. Obviously higher the coverage better the algorithm.

$$\text{coverage} = \frac{\text{Nbr of queries for which recommendations can be generated}}{\text{Total number of queries}}$$

Impression is defined as number of times recommendation was displayed and is analogous to surface rate of a feature.

4.2. Click through rate (CTR)

In the *online* setting from search-log one can measure number of times users are clicking on the suggestions. More precisely click through ratio is defined as

$$\text{CTR} = \frac{\text{Number of impressed suggestion which got atleast 1 click}}{\text{Number of impression}}$$

4.3. Normalized discounted cumulative gain

Discounted cumulative gain was first introduced by Järvelin and Kekäläinen (2000). DCG measures the usefulness, or gain, of a document based on its position in the result list. The gain is accumulated from the top of the result list to the bottom with the gain of each result discounted at lower ranks DCG makes following two assumptions: (1) highly relevant suggestions are more valuable to the user than marginally relevant suggestions, and (2) As the rank of the suggestion increases the value of the suggestion to the user decreases. Hence it provides as a good measure to evaluate relevance of an ordered list.

4.4. Runtime

Total time to generate suggestions for queries is critical for query suggestion algorithm to be adopted in practice. The runtime is divided in two parts: (a) Preprocessing time or model building, and (b) online suggestion generation or processing required for streaming queries.

Algorithms provided in Reda et al. (2012) and Al Hasan et al. (2011) are the algorithms which have online testing. Both these works in the pre-processing stage build the recommendations for queries and store them in a dictionary to be looked up when request from search engine arrive. Making the processing required for streaming query trivial.

5. Properties of large scale data

In practice the query suggestion is trained on search-log obtained from the same search engine. The biases introduced while collection of the log become a critical issue to be handled. Moreover the *query-log* data generated has long tail. And as more data is added to train the algorithms more dimensionality are also brought in. Two month of session log and queries occurring on www.ebay.com are used for this section.

5.1. Long tail

The session logs are used for all of the behavioral features in the algorithm. Figure 9 shows that the session length follow power law in both measures, namely page visited and number of searches.

The frequency of e-commerce queries follow a power-law distribution (Parikh and Sundaresan, 2008). As a consequence, high frequency queries (also called “head query” in the industry) have extensive footprint in the search log and it is easy to find a set of high quality recommendation for these queries using session-based co-occurrence. On the contrary, the tail (low frequency) queries appear sporadically in the query log, so it is difficult to find good recommendations for those from session-based co-occurrence. But, it is the tail queries for which the suggestions are highly useful to the site visitors.

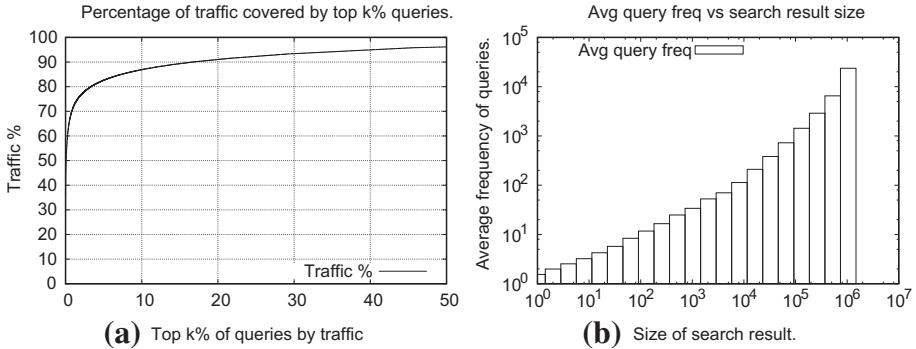


Fig. 8. (a) Long tail distribution of eBay search queries. 10% of the most frequent queries constitute about 87% of the search query traffic. (b) Relation between query frequency and search result-set size. It can be easily seen that the set of queries having weekly frequency value less than 10, return, on an average less than 100 items.

In Fig. 8a we show part of the long tail distribution; clearly 10% of the most frequent queries constitute about 87% of the search query traffic. For 20% of the most frequent queries cover almost 91% of the search traffic and for 50% of the most frequent queries, the coverage is about 96% of the search traffic. It clearly indicates that more than 50% of distinct queries all together contribute to less than 5% of the query traffic. For all these queries session-based data is sparse.

One may argue that if 90% coverage is obtained from less than 20% of queries, why do we care about the remaining 80% queries that form the long tail. In some sense, we could just deal with the head queries and forget about the rest. The tail availability boosts head sales by offering consumers the convenience of *one-stop shopping* for both their mainstream and niche interests. The long tail opportunities are nicely summarized in Goel et al. (2010).

Even the behavioral data collected on queries in query log follow a near-power law distribution. Figure 9a describes percentage of search traffic and percentage of unique queries as a function of number of clicks received. Figure 9b describes the same quantities in cumulative fashion. It can easily be observed that roughly 50% of searches receive less than four clicks. 80% of unique queries receive less than two clicks. Similar long tail is present with respect to user browsing activity and searches within session. Figure 9c and d displays the power law in user session activity.

Long tail queries make it hard to build off-line suggestions for all user queries. Typically, we mine query logs for a certain time period to build the query suggestion index (a hashtable, where the key is the user query and the value is the set of suggestion queries). If a query does not appear at all in this time period, no suggestions can be built for it. Our study on eBay query log reveals this problem. We collected all the distinct query strings for two consecutive days. Around 30% of the queries from the first day also appear on the next day. So, an index built using the first day log (we refer to it as training data) would have suggestion for only 30% of the distinct user queries executed on the second day. If we consider more days of training data to build the index, the index can cover more user queries. This is shown in Fig. 10a, where we increase the query log window from one day to

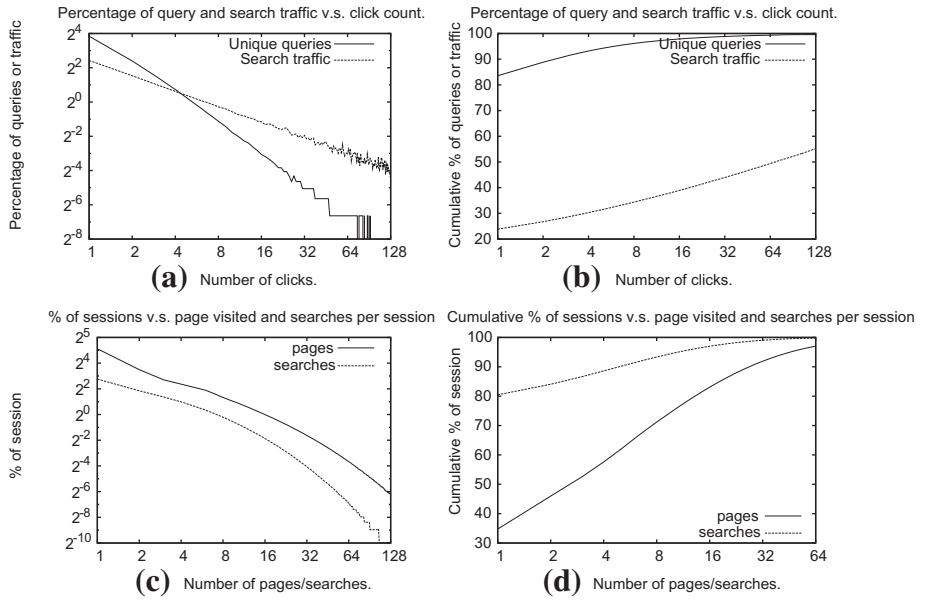


Fig. 9. (a) Percentage of unique queries and search traffic w.r.t. number of clicks per query. 18% of searches and 69% of unique queries have 0 clicks. (b) Cumulative percentage of unique queries and search traffic w.r.t. number of clicks per query. 30% of searches and 80% of unique queries have four or less clicks. (c) Fraction of sessions w.r.t. number of page visited compared to number searches in them. (d) Cumulative fraction of sessions w.r.t. number of page visited compared to number searches in them.

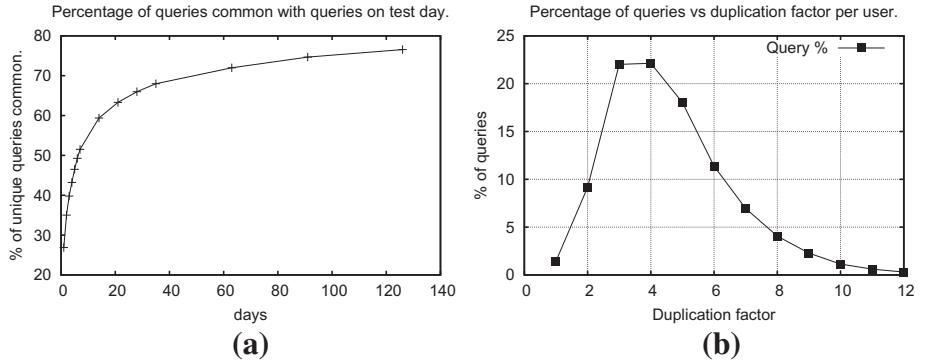


Fig. 10. (a) Query overlap over several days for queries from www.ebay.com. With historical data being pushed till 120 days the overlap only went till as high as 77%. This serves as a good upper bound on coverage for lot of algorithms. (b) Query duplication factor per query per user. It is different for different queries.

3 months and show the percentage of user queries (on a day outside these training days) that is available in the index. However, increasing the training data arbitrarily has adversarial affect on the quality of the suggestions. For example, buzz queries (Parikh and Sundaresan, 2009) (such as, world cup soccer 2010) or seasonal queries

(such as Halloween, Christmas) could surface as a query suggestion during the wrong time period (after the event is gone or in the off-season). More importantly, for tail queries the suggestion might be good only for a certain period of time when the corresponding item is still available in the inventory. Otherwise, the suggestions can lead to no results causing user frustration.

5.2. Bias

Query logs collected by various websites, such as Google, Yahoo, Bing, and eBay, are generated by searches being performed on the search engine. In addition to humans performing searches, other sources that perform searches include bots or robots, API's used by external websites and programs, various add-ons and extensions installed in the browser, links and widgets embedded in website, for example, affiliate sites and social sites, such as twitter and facebook. **Bots** are programs that automatically issue queries and clicks to the search engine. Search assistive tools, such as query suggestion, are intended for humans, so activities of bots in the log are noise and they compromise the quality of the query suggestion index. In the web search domain, there are many studies (Kang et al., 2010; Cooley et al., 1999; Sadagopan and Li, 2008) that analyze the influence of bots and suggest mechanisms to remove bot queries from the query logs. The search logs are corrupted by unexpected queries or atypical frequencies of queries or advanced queries that humans are unlikely to type in. Aggressive bots perform extremely long search sessions with no contextual similarity. We observed search sessions (duration of each session is, typically, half an hour) with more than ten thousand searches in the query logs. Though these are easy to tag, there are bots that perform small number of searches in a session; these are hard to catch. Widgets and embedded links also generate large quantity of searches which look perfectly legitimate. In many cases, these are hard to tag and remove.

We also observed unique users performing multiple searches with the same seed query. Pagination or research is most prevalent reason for performing searches with same queries. But, this factor is different for different queries. From Fig. 10b, we can see that 60% of the queries have four or less instances per user per week, whereas remaining 40% has more than four. The skewness in query multiplicity rate can adversely affect the query co-occurrence statistics. For tail queries, where the actual frequency is low this can give rise to sub-optimal or poor query suggestion. For example, training on the same data set and the same algorithm, the suggestion set can vary based on whether we consider the raw association count or the distinct user association count. We show an example in Table 2 which shows the suggestion-list for the query “calculator.” It is easy to see that the suggestions from the second case are better; “fluke” and “pocket knife” are not attractive suggestions for the query “calculator” and those have been thrown away when the query association data are de-duplicated by the distinct users.

Normalization by the distinct user count is just a form of cleaning (noise removal). But, one can use much more aggressive cleaning techniques, such as those suggested in Cooley et al. (1999) and Sadagopan and Li (2008)). Rule-based cleaning techniques can also be applied to clean the data; for example, a simple rule can be used to ignore sessions without purchase events. The logic behind this is that the

Table 2
Session co-occurrence data for the query *calculator*

Raw co-occurrence data	De-duplicated co-occurrence data
fluke, texas instrument, adding machine, pocket knife, ti 84, scientific calculator	casio calculator, scientific calculator, calculator hp, texas instrument, ti84

query associations learnt from the successful user sessions would be much cleaner. But too aggressive cleaning of query logs can lead to loss of information and can be damaging for the tail queries. This phenomenon is identical to the precision-recall tradeoff; aggressive cleaning can lead to more precise suggestion, but at the same time the recall dips sharply (we may not have any suggestions for many queries).

Users are now performing their searches from different **platforms**. Web browser on PCs, **smart phones**, and **tablets** (apps and browser). Each platform has unique screen size limitation and forces the website to provide different browsing experience. These different browsing experience then result in different searching behavior. Normalizing or accounting for these changes then become vital for correctness of behavioral features of the algorithm.

6. Query Suggestion in practice

In this chapter we discuss some of the issues to be considered while building a query suggestion algorithm at webscale. [Al Hasan et al. \(2011\)](#), [Parikh and Sundaresan \(2008\)](#), [Reda et al. \(2012\)](#) and [Beeferman and Berger \(2000\)](#)) are few of such paper which describe query suggestion in practice.

6.1. When is query suggestion useful?

As discussed in Section 1 query suggestion is a useful tool for users to explore or short-circuit their search tasks. Figure 11a shows click through performance of query suggestion on eBay search engine with respect to the size of the result set of the query. One can clearly observe that query suggestion is twice as clicked upon queries with less than 20 matching items (urls) than queries more than 100 items. Figure 11b compares the click through ratio of related vs refinement suggestion. It can be observed that the usage of refinement or related suggestion can vary by a factor of 3 depending on theme of original query.

For instance, the set of queries having weekly frequency value less than 10, return, on an average less than 100 items. This low recall, typically, motivates the searchers to use the assistive tools available on the site; Fig. 11a shows evidence in favor of this claim. The related search click-through rate rises sharply as the recall values fall below 200. The CTR for recall value less than 10 is almost double than that for the recall value of 100. For long tail queries the searchers need the most assistance, but unfortunately for those queries the session log contain sparse information to be able to help with query suggestion. [Goel et al. \(2010\)](#) not only shows that the long tail brings in opportunities, but also clearly reveals that it also brings unique challenges to build machine-learned systems for query suggestion, clustering, or ranking.

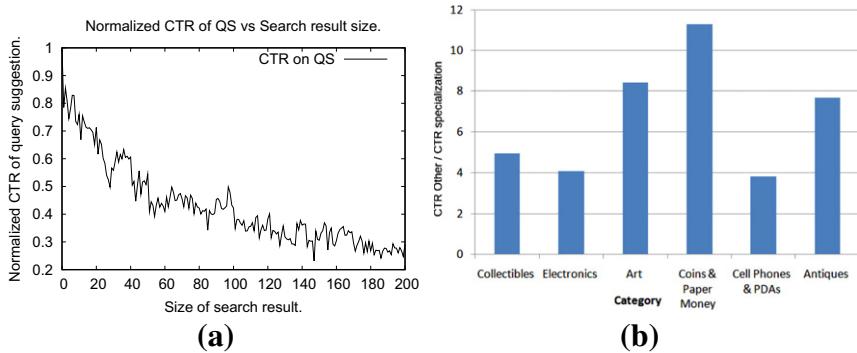


Fig. 11. (a) CTR of Query Suggestions with the result set size of the search. QS is clicked twice as often on searches which have less than 20 matching items as compared to searches with more than 100 items. (b) CTR ratios for other task v.s. specialization suggestions across different categories. This can vary by a factor of 3 from category to category.

6.2. Addressing practical data challenges

In this section we summarize various strategies used to overcome challenges that real data proposes.

6.2.1. Long tail

Section 5 discusses the challenges of long tail queries in detail. In brief, searches and queries and behavioral data follow power law and have long tail. CTR measurement showed that the query suggestion feature is used more often on long tail queries than head queries. 18% of searches and 69% of unique queries have 0 clicks. These queries will never appear in the query-click graphs or in the session-based similarity. Approaches which use textual similarity or content-based similarity may still be able to generate similarity value between two queries.

Table 3 shows index size increases almost linearly with the amount of data, but impression rate (normalized data against the baseline) does not improve linearly; for phase 1 and phase 2 indexes, we only have 12% and 22% improvement over the baseline.

6.2.2. Quality of data

Bot-bias, user-bias, and platform bias are three major issues with data quality. Data preparation strategy should involve bot-filtering. Bots and addicted user are few of the biggest outliers in the behavioral data. We have observed some of the session length greater than 1000. One such session can hijack features for some queries, especially in the tail and torso.

Approaches like textual similarity are not much affected by many of these biases. To generate high quality behavioral similarity measures either (a) remove these outliers from the data set (b) User deduplication has to be carried out. One user one vote limits the effect of outlier sessions on features. Table 3 shows that the phase 1 and the phase 2 index have a CTR improvement by 13% and 41% with respect

to the baseline. It proves that both data cleaning and suggestion refinements are important.

6.2.3. Tractability of algorithm

Amount of time to generate query suggestion is broken down into offline (pre-processing) time and on-line (when query arrives) time. Most of the strategies pre-compute suggestion list and cache them in dictionaries; thus making on-line time extremely low. This tradeoff does lead to coverage loss as there can be no suggestion for queries which are not in the cache. Hence tractability discussion is restricted to offline processing. [Hadoop. Apache \(2008\)](#) is used to do parallel computation of similarity scores.

Many textual similarity scores dictate computation of similarity between $\Omega(n^2)$ query pairs; where n is number of queries which can easily be as big as 100 million. [Singh et al. \(2011a\)](#) compute textual features between query pair which co-occur in session, reducing possible pairs from quadrillion to few billion.

Query log-based similarity scores require graph operation of 100 million (queries) by 1 billion (URLs) bipartite graph. Random walk, computation of hitting time, *tf-idf* over neighborhood, agglomerative clustering becomes tractable by converting them to iterable approximations. Where each iteration can be carried out using map-reduce.

Session-based similarity scores are tractable as each session has to be processed independently to generate respective feature counts. Which then later on get aggregated and deduped resulting into compact vectors.

6.2.4. Presentation

In real-world online testing scenarios, presentation of query suggestion has also been observed to be highly important. Table 3 shows that the presentation-change has a CTR improvement by 13% with respect to the baseline. The presentation-change comprised of highlighting the new words in the suggestion when compared to original query. [Zha et al. \(2009\)](#) show that visual query suggestion is able to more precisely and more quickly help users specify and deliver their search intents.

7. Closing remarks

In this chapter we described the design, implementation, and evaluation aspects of Query Suggestion, an integral module for most search engines. Our emphasis was on highlighting the strengths and weaknesses of different approaches, illustrating the nature of large scale data , pointing out the challenges in building such systems and describing ways to address some of those challenges in practice. Query Suggestions is a ripe area of research. In addition to the methods described in this chapter, there has been active work ranging all the way from using temporal signals ([Chien and Immorlica, 2005; Parikh et al., 2012](#)) to compute Query Suggestions, using machine learning methods to rank query suggestion candidates ([Ozertem et al., 2012](#)) to building optimizing frameworks for Query Suggestions ([Anagnostopoulos et al., 2010](#)). We hope that the material presented in this chapter is broad and detailed

Table 3

Results from online experiments as shown by Al Hasan et al. (2011). The approach used by them is ensemble with textual similarity, session-based query short-cutting similarity and query long-based similarity. The index for presentation-change is identical to the baseline. The only change there is in UI. The non-common part of recommend query with the original query is highlighted and boldened

Evaluation Metrics	Indexes			
	Baseline	Presentation-change	Phase 1	Phase 2
Data	10 days	10 days	20 days	30 days
Bot removal	No	No	Yes	Yes
User deduping	No	No	Yes	Yes
Presentation	Basic	Advanced	Basic	Basic
Index size	4.8 millions	4.8 millions	9.8 millions	22.6 millions
Impression rate	1	1	1.12	1.22
CTR	1	1.13	1.12	1.41

enough for readers to be able to appreciate the need for efficient Query Suggestion systems in practice and the difficulties associated with building such systems. Some of the approaches we described to cope with large scale data biases, long tail sparsity, and processing as well as run-time efficiencies are not strongly tied only to Query Suggestion systems and should be useful in the design of various other scalable search engine features.

References

- Al Hasan, Mohammad, Parikh, Nish, Singh, Gyanit, Sundaresan, Neel, 2011. Query suggestion for e-commerce sites. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM '11. ACM, New York, NY, USA, pp. 765–774.
- Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., 2010. An optimization framework for query recommendation. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining. ACM, pp. 161–170.
- Baeza-Yates, Ricardo, Tiberi, Alessandro, 2007. Extracting semantic relations from query logs. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '07. ACM, New York, NY, USA, pp. 76–85.
- Baeza-yates, Ricardo, Hurtado, Carlos, Mendoza, Marcelo, 2004. Query recommendation using query logs in search engines. In: International Workshop on Clustering Information Over the Web (ClustWeb, in conjunction with EDBT), Crete. Springer, pp. 588–596.
- Baraglia, Ranieri, Castillo, Carlos, Donato, Debora, Nardini, Franco Maria, Perego, Raffaele, Silvestri, Fabrizio, 2009. Aging effects on query flow graphs for query suggestion. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, Hong Kong, China. ACM, New York, NY, USA, pp. 1947–1950.
- Beferman, Doug, Berger, Adam, 2000. Agglomerative clustering of a search engine query log. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '00. ACM, New York, NY, USA, pp. 407–416.
- Bhatia, Sumit, Majumdar, Debapriyo, Mitra, Prasenjit, 2011. Query suggestions in the absence of query logs. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11. ACM, New York, NY, USA, pp. 795–804.
- Boldi, Paolo, Bonchi, Francesco, Castillo, Carlos, Donato, Debora, Vigna, Sebastiano, 2009. Query suggestions using query-flow graphs. In: Proceedings of the 2009 Workshop on Web Search Click Data, WSCD '09. New York, NY, USA. ACM, pp. 56–63.

- Cao, Huanhuan, Jiang, Dixin, Pei, Jian, He, Qi, Liao, Zhen, Chen, Enhong, Li, Hang, 2008. Context-aware query suggestion by mining click-through and session data. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08. ACM, New York, NY, USA, pp. 875–883.
- Chatzopoulou, G., Eirinaki, M., Koshy, S., Mittal, S., Polyzotis, N., Varman, J.S.V., 2011. The querie system for personalized query recommendations. *IEEE Data Eng. Bull.* 34 (2), 55–60.
- Chien, Steve, Immorlica, Nicole, 2005. Semantic similarity between search engine queries using temporal correlation. In: Proceedings of the 14th International Conference on World Wide Web, WWW '05. ACM, New York, NY, USA, pp. 2–11.
- Cooley, Robert, Mobasher, Bamshad, Srivastava, Jaideep, 1999. Data preparation for mining world wide web browsing patterns. *Knowl. Inf. Syst.* 1, 5–32.
- Craswell, Nick, Szummer, Martin, 2007. Random walks on the click graph. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07. ACM, New York, NY, USA, pp. 239–246.
- Cronen-Townsend, Steve, Croft, Bruce, W., 2002. Quantifying query ambiguity. In: Proceedings of the Second International Conference on Human Language Technology Research, HLT '02. Morgan Kaufmann Publishers Inc., pp. 104–109.
- Frants, Valery, I., Shapiro, Jacob, Taksa, Isak, Voiskunki, Vladimir, G., 1999. Boolean search: Current state and perspectives. *J. Am. Soc. Inf. Sci.* 50 (1), 86–95.
- Gilleland, M. et al., 2009. Levenshtein distance, in three flavors. Merriam Park Software: <<http://www.merriampark.com/ld.htm>>.
- Goel, Sharad, Broder, Andrei, Gabrilovich, Evgeniy, Pang, Bo, 2010. Anatomy of the long tail: Ordinary people with extraordinary tastes. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10. ACM, New York, NY, USA, pp. 201–210.
- Hadoop. Apache, 2008. <<http://hadoop.apache.org/>>.
- Hearst, Marti, A., 2009. Search User Interfaces, first ed. Cambridge University Press, New York, NY, USA.
- Järvelin, Kalervo, Kekäläinen, Jaana, 2000. Ir evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '00. ACM, New York, NY, USA, pp. 41–48.
- Jones, Rosie, Fain, Daniel, C., 2003. Query word deletion prediction. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '03. ACM, New York, NY, USA, pp. 435–436.
- Kang, Hongwen, Wang, Kuansan, Soukal, David, Behr, Fritz, Zheng, Zijian, 2010. Large-scale bot detection for search engines. In: Proceedings of the 19th International Conference on World wide web, WWW '10. ACM, New York, NY, USA, pp. 501–510.
- Kato, Makoto, P., Sakai, Tetsuya, Tanaka, Katsumi, 2012. Structured query suggestion for specialization and parallel movement: effect on search behaviors. In: Proceedings of the 21st International Conference on World Wide Web, WWW '12. ACM, New York, NY, USA, pp. 389–398.
- Ma, H., Lyu, M.R., King, I., 2010. Diversifying query suggestion results. In: Proceedings of AAAI, vol. 10.
- Mei, Qiaozhu, Zhou, Dengyong, Church, Kenneth, 2008. Query suggestion using hitting time. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08. ACM, New York, NY, USA, pp. 469–478.
- Metzler, Donald, Dumais, Susan, Meek, Christopher, 2007. Similarity measures for short segments of text. In: Proceedings of the 29th European Conference on IR Research, ECIR'07. Springer-Verlag, Berlin, Heidelberg, pp. 16–27.
- Ozertem, Umut, Chapelle, Olivier, Donmez, Pinar, Velipasaoglu, Emre, 2012. Learning to suggest: a machine learning framework for ranking query suggestions. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12. ACM, New York, NY, USA, pp. 25–34.
- Parikh, Nish, Sundaresan, Neel, 2008. Inferring semantic query relations from collective user behavior. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08. ACM, New York, NY, USA, pp. 349–358.

- Parikh, Nish, Sundaresan, Neel, 2009. Buzz-based recommender system. In: Proceedings of the 18th International Conference on World Wide Web, WWW '09. ACM, New York, NY, USA, pp. 1231–1232.
- Parikh, Nish, Hong, Hui, Chiu, Evan, Neel Sundaresan, 2012. Mining large scale temporal dynamics with hadoop. In: Hadoop Summit.
- Reda, Azarias, Park, Yubin, Tiwari, Mitul, Posse, Christian, Shah, Sam, 2012. Metaphor: a system for related search recommendations. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM '12. ACM, New York, NY, USA, pp. 664–673.
- Baeza-Yates, Ricardo, A., Ribeiro-Neto, Berthier, 1999. Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- Sadagopan, Narayanan, Li, Jie, 2008. Characterizing typical and atypical user sessions in clickstreams. In: Proceedings of the 17th International Conference on World Wide Web, WWW '08. ACM, New York, NY, USA, pp. 885–894.
- Sahami, Mehran, Heilman, Timothy, D., 2006. A web-based kernel function for measuring the similarity of short text snippets. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06. ACM, New York, NY, USA, pp. 377–386.
- Shi, Xiaodong, Yang, Christopher, C., 2006. Mining related queries from search engine query logs. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06. ACM, New York, NY, USA, pp. 943–944.
- Silverstein, Craig, Henzinger, Monika, Marais, Hannes, Moricz, Michael, 1998. Analysis of a very large altavista query log.
- Singh, Gyanit, Parikh, Nish, Sundaresan, Neel, 2011a. Query suggestion at scale. In: Hadoop Summit.
- Singh, Gyanit, Parikh, Nish, Sundaresn, Neel, 2011b. User behavior in zero-recall ecommerce queries. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11. ACM, New York, NY, USA, pp. 75–84.
- Song, Ruihua, Luo, Zhenxiao, Wen, Ji-Rong, Yu, Yong, Hon, Hsiao-Wuen, 2007. In: Proceedings of the 16th international conference on World Wide Web, WWW '07. ACM, New York, NY, USA, pp. 1169–1170.
- Sydow, Marcin, Ciesielski, Krzysztof, Wajda, Jakub, 2012. Introducing diversity to log-based query suggestions to deal with underspecified user queries. In: Proceedings of the 2011 International Conference on Security and Intelligent Information Systems, SIIS'11. Springer-Verlag, Berlin, Heidelberg, pp. 251–264.
- White, Ryen, W., Huang, Jeff, 2010. Assessing the scenic route: measuring the value of search trails in web logs. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10. ACM, New York, NY, USA, pp. 587–594.
- Zha, Zheng-Jun, Yang, Linjun, Mei, Tao, Wang, Meng, Wang, Zengfu, 2009. Visual query suggestion. In: Proceedings of the 17th ACM International Conference on Multimedia, MM '09. ACM, New York, NY, USA, pp. 15–24.
- Zhang, Zhiyong, Nasraoui, Olfa, 2006. Mining search engine query logs for query recommendation. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06. ACM, New York, NY, USA, pp. 1039–1040.
- Zhu, X., Guo, J., Cheng, X., 2010. Recommending diverse and relevant queries with a manifold ranking based approach. In: SIGIR.10 Workshop on Query Representation and Understanding.

Subject Index

A

Algorithm

- basic solution, 70
- dictionary learning, 211
- for iris images, estimation, 393
- performance, data dependency of, 285
- selection, 450

Arabic documents, manifold learning for shape-based

- recognition of historical, 471
- experimental results, 481
 - PCA, 484
 - PCA+GDA, 485
 - PCA+LLE, 485
- feature extraction, 480
 - Gray stroke map (GSM), 481
- manifold learning, 475
 - generalized discriminant analysis (GDA), 478
 - Locally linear embedding (LLE), 475
 - measure of manifold quality, 477
 - Supervised locally linear embedding (SLLE), 476
- problem statement, 475

Arabic text recognition, machine learning in

- handwritten, 443
- Arabic script, recognition challenges, 444
- features for text recognition, 455
 - chain code and critical point features, 456
 - directional profile features, 457
 - discrete symbol features, 458
 - feature selection and combination, 459
 - gradient-structural-concavity (GSC) features, 455
- learning paradigms, 447
 - co-training, 449
 - oracle training, 451
 - selection algorithm, 450
 - structural learning, 452
- models for recognition, 460
 - Artificial Neural Network (ANN), 462
 - Deep Belief Networks (DBN), 463

— ensemble of learners, 460

— Hidden Markov Models (HMM), 466

— Restricted Boltzmann machines (RBMs), 465

Artificial Neural Network (ANN), 462

B

Bagging, boosting, and random forests using R, 101

- bagging, 107, 115
 - training set, 108
- boosting, 113, 115
 - data (training sample), 113
 - boosting using R package, ada, 144
 - classification tree, 116
 - *versus* logistic regression, 126
 - properties, 117
 - data sets and rationale, 103
 - Boston housing, 106
 - Breast Cancer data of University of Wisconsin hospitals, 104
 - diabetes data of Pima Indians, 105
 - heart data of San Diego Medical Center, 103
 - Ozone data, 106
 - Soybean data, 106
 - spliced DNA sequence data of Genbank, 105

bagging, boosting, and random forests using R

- logistic regression, classification tree *versus*, 126
- random forest, 127
 - discussion of inputs, 128
 - genetics, and cross-validation, 134
 - regression trees, 139

Bayesian graphical models for iris recognition, 390

- application of, 381
- correlation filter-based iris matching, 386
- estimation algorithm for iris images, 393
- Gabor wavelet-based matching
 - iris encoding, 385
 - iris matching, 386
 - iris segmentation, 384
 - iris unwrapping, 384
- MAP estimation of transformation, 392
- numerical results, 395

- Biometric matching systems, 156
 - Boston housing, 106
 - Breast Cancer data of University of Wisconsin hospitals, 104
- C**
- Cambridge hand gesture dataset, 319
 - analysis, 321
 - simulation result, 320
 - Classification tree, 116
 - *versus* logistic regression, 126
 - properties, 117
 - Click through rate (CTR), 508
 - Conditional Random Fields (CRF)
 - implementations of CRF scene labelings
 - hierarchical CRF for scene labeling, 242
 - inference, 232
 - Markov Chain Monte Carlo (MCMC), 233
 - for scene labeling, 227, 229
 - definition, 231
 - implementations of CRF scene labelings
 - fully connected CRF for scene labeling, 243
 - scene parsing, 236
 - color, 237
 - datasets, 239
 - end-to-end example of CRF scene labeling implementation, 240
 - location, 239
 - scene features, 237
 - texture, 239
 - undirected graphical models, 229
 - undirected graphical models for labeling, 231
 - variational mean field, 234
 - Continuous optimization, 50
 - maximum likelihood optimization, 53
 - optimal control, 51
 - Correlation filter-based iris matching, 386
 - Cross-entropy method for estimation, 19
 - estimation setting, 20
 - extensions
 - root finding, 32
 - sampling directly from zero-variance distribution, 26
 - transformed likelihood ratio (TLR), 29–30
 - Cross-entropy method for optimization, 35
 - applications to combinatorial optimization, 40
 - Boolean satisfiability problem, 42
 - knapsack packing problem, 40
 - network planning problem, 44
 - permutation Monte Carlo and merge process, 46
 - sampling with budget constraint, 48
 - continuous optimization, 50
 - maximum likelihood optimization, 53
 - optimal control, 51
 - from estimation to optimization, 36
 - cross-entropy for rare-event probability estimation, 36
 - cross-entropy method for optimization, 38
 - Cross-media relevance model (CMRM), 256
- D**
- Data, query suggestion with large scale, 493
 - approaches to generation of query suggestions, 501
 - content-based similarity, 503
 - hitting time, 506
 - query adjacency, 506
 - query-clicked URL set based similarity, 504
 - query-log based similarity, 504
 - query short-cutting, 507
 - random walks, 505
 - session-based similarity, 506
 - shortest-path and distance as similarity, 505
 - textual similarity among queries, 501
 - evaluation methods of QS, 508
 - Click through rate (CTR), 508
 - coverage, 508
 - impression, 508
 - normalized discounted cumulative gain, 509
 - runtime, 509
 - practice, query suggestion in, 513
 - addressing practical data challenges, 514
 - long tail, 514
 - quality of data, 514
 - properties of large scale data, 509
 - bias, 512
 - long tail, 509
 - terminology, 499
- Deep Belief Networks (DBN), 463
- Diabetes data of Pima Indians, 105
- Dictionary-based methods for object recognition, 203
 - dictionary learning, 210
 - dictionary learning algorithms, 211
 - dictionary learning from partially labeled data, 220
 - discriminative dictionary learning, 213
 - joint dimensionality reduction and dictionary learning, 218
 - non-linear kernel dictionary learning, 216
 - unsupervised dictionary learning, 220
 - sparse representation, 204
 - robust biometrics recognition via sparse representation, 208
 - sparse representation-based classification, 206
- Document layout analysis, learning algorithms for, 401
 - classification levels in layout analysis, 403
 - connected component classification, 409

- functional labeling, 415
- layout analysis approaches, 402
- performance evaluation, 403
- pixel classification, 405
- region classification, 412
- text region segmentation, 411
- zone classification, 406

F

Fusion in matching systems, selected approaches to, 158

- dynamic classifier selection, 159
- fusion rules and score normalization, 158
- user-and template-specific combinations, 160
- utilizing external knowledge, 161

G

Gabor wavelet-based matching, 383

- iris encoding, 385
- iris matching, 386
- iris segmentation, 384
- iris unwrapping, 384

GDA. *See* Generalized discriminant analysis (GDA)

Generalized discriminant analysis (GDA), 478

Gradient-structural-concavity (GSC) features, 455

Graphical models

- undirected, 229
- for labeling, 231

Gray stroke map (GSM), 481

H

Handwriting images, serialization of, 423

- sliding window feature extraction, 425
- text line normalization, 424

Handwritten word recognizers, 157

Heart data of San Diego Medical Center, 103

Hidden Markov Models (HMM), 466

- HMM-based text line recognition, 426
 - confidence models, 433
 - hidden Markov models, 427
 - language models, 431
- for off-line cursive handwriting recognition, 421
- serialization of handwriting images, 423
 - sliding window feature extraction, 425
 - text line normalization, 424

HMM. *See* Hidden Markov Models (HMM)

I

Iris recognition, Bayesian graphical models for, 390

- application of, 381
- correlation filter-based iris matching, 386
- estimation algorithm for iris images, 393
- Gabor wavelet-based matching
 - iris encoding, 385

- iris matching, 386
- iris segmentation, 384
- iris unwrapping, 384
- MAP estimation of transformation, 392
- numerical results, 395

K

KTH human action dataset, 314

- simulation result, 315

L

LAPD solutions, 313

Learning algorithms

- dictionary, 211
- for document layout analysis, 401
 - classification levels in layout analysis, 403
 - connected component classification, 409
 - functional labeling, 415
 - layout analysis approaches, 402
 - performance evaluation, 403
 - pixel classification, 405
 - region classification, 412
 - text region segmentation, 411
 - zone classification, 406

Linear Discriminant Analysis (LDA), 253

LLE. *See* Locally linear embedding (LLE)

Locally linear embedding (LLE), 475

Logistic regression, classification tree *versus*, 126

Luminance differential trajectory and aligned

- projection distance, video activity recognition by, 302
- Cambridge hand gesture dataset, 319
 - analysis, 321
 - simulation result, 320
- DLFT solutions, 313
- experiments, 313
- KTH human action dataset, 314
 - simulation result, 315
- LAPD solutions, 313
- problem formulation, 305
 - differential luminance field trajectory approach, 311
 - dimensionality reduction, 307
 - luminance aligned projection distance approach, 309
 - maximum likelihood detection, 308
 - video representation, 307
- related work, 303
- Youtube action dataset, 321
 - simulation result, 322

M

Machine learning in handwritten Arabic text

recognition, 443

- Arabic script, recognition challenges, 444

- features for text recognition, 455
 - chain code and critical point features, 456
 - directional profile features, 457
 - discrete symbol features, 458
 - feature selection and combination, 459
 - gradient-structural-concavity (GSC) features, 455
- learning paradigms, 447
 - co-training, 449
 - oracle training, 451
 - selection algorithm, 450
 - structural learning, 452
- models for recognition, 460
 - Artificial Neural Network (ANN), 462
 - Deep Belief Networks (DBN), 463
 - ensemble of learners, 460
 - Hidden Markov Models (HMM), 466
 - Restricted Boltzmann machines (RBMs), 465
- Manifold learning for shape-based recognition of historical Arabic documents, 471
- experimental results, 481
 - PCA, 484
 - PCA+GDA, 485
 - PCA+LLE, 485
- feature extraction, 480
 - Gray stroke map (GSM), 481
- manifold learning, 475
 - generalized discriminant analysis (GDA), 478
 - Locally linear embedding (LLE), 475
 - measure of manifold quality, 477
 - Supervised locally linear embedding (SLLE), 476
- problem statement, 475
- Markov Chain Monte Carlo (MCMC), 233
- Masquerade detection, user behavior monitoring and profiling scheme for, 354
- data collection, 361
 - data sanitization, 362
 - issues with data collection and use, 362
- experimental design, 367
 - classification rate issues, 376
 - comprehensive data, 370
 - mouse data, 368
- feature extraction, 363
 - design of feature extraction engine, 366
 - feature vector generation, 367
 - sliding window protocol, 365
- feature selection methods, 363
- Masquerade detection based on GUI data, 358
- previous anomaly/masquerade detection efforts, 357
- Support Vector Machines, 358
 - advantages of SVM, 361
 - Linear SVM, 359
 - nonlinear SVM, 360
- Matching score fusion methods, 151
 - classifier ensembles, 154
 - complexity types of classifier combination methods, 163
 - matching systems, 156
 - biometric matching systems, 156
 - handwritten word recognizers, 157
 - modeling matching score dependencies, 165
 - operating modes of matching systems, 162
 - performance of classifiers and their combination, 152
 - score combination applications
 - multiple samples, 168
 - performance prediction, 168
 - verification and identification systems, 167
 - selected approaches to fusion in matching systems, 158
 - dynamic classifier selection, 159
 - fusion rules and score normalization, 158
 - user- and template-specific combinations, 160
 - utilizing external knowledge, 161
 - statistical learning for classifier combinations, 153
- O**
- Object recognition, dictionary-based methods for, 203
 - dictionary learning, 210
 - dictionary learning algorithms, 211
 - dictionary learning from partially labeled data, 220
 - discriminative dictionary learning, 213
 - joint dimensionality reduction and dictionary learning, 218
 - non-linear kernel dictionary learning, 216
 - unsupervised dictionary learning, 220
 - sparse representation, 204
 - robust biometrics recognition via sparse representation, 208
 - sparse representation-based classification, 206
- Ozone data, 106
- P**
- Probability collectives in optimization, 61
 - delayed and immediate sampling PC, 62
 - delayed sampling experiments
 - annealing, 73
 - basic solution algorithm, 70
 - collectives for control, 78
 - continuous problem, example for, 73
 - data aging, 71
 - discrete optimization, 77
 - modifications for continuous variables, 71
 - performance on larger-scale problems, 76
 - regressions, 72

- delayed sampling theory
 - Brouwer updating, 67
 - gradient of free energy, 66
 - Maxent Lagrangian, 64
 - Monte Carlo-based gradient descent and shrink-wrapping, 67
 - nearest Newton, 69
 - private utilities, 69
 - immediate sampling experiments, 85
 - bagging, 93
 - constant β , 89
 - cross-validation for regularization and model selection, 94
 - cross-validation to schedule β , 90
 - Gaussian densities, 86
 - immediate sampling with single Gaussian, 87
 - implementation details, 88
 - minimizing pq KL distance, 86
 - mixture models, 87
 - other applications of PL techniques to immediate sampling, 96
 - quadratic G(x), 88
 - varying β , 90
 - immediate sampling theory, 82
 - Monte Carlo optimization, 82
 - naive immediate sampling, 83
 - parametric machine learning, 84
 - PL equals MCO, 85
 - intuition behind probability collectives, 61
 - relevant literature and roadmap of paper, 63
-
- Q**
 - Query suggestion with large scale data, 493
 - approaches to generation of query suggestions, 501
 - content-based similarity, 503
 - hitting time, 506
 - query adjacency, 506
 - query-clicked URL set based similarity, 504
 - query-log based similarity, 504
 - query short-cutting, 507
 - random walks, 505
 - session-based similarity, 506
 - shortest-path and distance as similarity, 505
 - textual similarity among queries, 501
 - evaluation methods of QS, 508
 - Click through rate (CTR), 508
 - coverage, 508
 - impression, 508
 - normalized discounted cumulative gain, 509
 - runtime, 509
 - practice, query suggestion in, 513
 - addressing practical data challenges, 514
 - long tail, 514
 - quality of data, 514
-
- properties of large scale data, 509
 - bias, 512
 - long tail, 509
 - terminology, 499
-
- R**
 - Random forest, 127
 - discussion of inputs, 128
 - genetics, and cross-validation, 134
 - Regression trees, 139
 - Restricted Boltzmann machines (RBMs), 465
 - Robust biometrics recognition via sparse representation, 208
-
- S**
 - Sampling experiments, probability collectives in optimization
 - delayed
 - annealing, 73
 - basic solution algorithm, 70
 - collectives for control, 78
 - continuous problem, example for, 73
 - data aging, 71
 - discrete optimization, 77
 - modifications for continuous variables, 71
 - performance on larger-scale problems, 76
 - regressions, 72
 - immediate, 85
 - bagging, 93
 - constant β , 89
 - cross-validation for regularization and model selection, 94
 - cross-validation to schedule β , 90
 - Gaussian densities, 86
 - immediate sampling with single Gaussian, 87
 - implementation details, 88
 - minimizing pq KL distance, 86
 - mixture models, 87
 - other applications of PL techniques to immediate sampling, 96
 - quadratic G(x), 88
 - varying β , 90
 - Sampling theory, probability collectives in optimization
 - delayed
 - Brouwer updating, 67
 - gradient of free energy, 66
 - Maxent Lagrangian, 64
 - Monte Carlo-based gradient descent and shrink-wrapping, 67
 - nearest Newton, 69
 - private utilities, 69
 - immediate, 82
 - Monte Carlo optimization, 82
 - naive immediate sampling, 83

- parametric machine learning, 84
- PL equals MCO, 85
- Scene labeling, CRF for, 227
 - CRF, 229
 - definition, 231
 - inference, 232
 - undirected graphical models, 229
 - undirected graphical models for labeling, 231
 - implementations of CRF scene labelings
 - fully connected CRF for scene labeling, 243
 - hierarchical CRF for scene labeling, 242
 - Markov Chain Monte Carlo (MCMC), 233
 - scene parsing, 236
 - color, 237
 - datasets, 239
 - end-to-end example of CRF scene labeling implementation, 240
 - location, 239
 - scene features, 237
 - texture, 239
 - variational mean field, 234
- Score combination applications
 - multiple samples, 168
 - performance prediction, 168
 - verification and identification systems, 167
- Sequential bootstrap, 4
 - bootstrapping empirical measures with random sample size, 9
 - convergence rates for sequential bootstrap, 12
 - resampling scheme, 7
 - second-order correctness of sequential bootstrap, 14
- Shape-based image classification and retrieval, 250
 - classification and retrieval models, 252
 - cross-media relevance model (CMRM), 256
 - Linear Discriminant Analysis (LDA), 253
 - maximum entropy, 255
 - Naïve Bayes, 252
 - retrieval, 257
 - support vector machines, 254
 - SVMs for binary classification, 254
 - SVMs for multiple classes, 254
 - classification experiments, 261
 - features, 257
 - centroid distance function, 258
 - fourier transform, 259
 - invariance, 260
 - preprocessing, 258
 - profile feature set, 258
 - multiple class labels, 265
 - prior work, 251
 - retrieval, 262
 - retrieval with COIL-100 database, 263
 - retrieval with MPEG-7 database, 262
- Soft biometrics for surveillance, 327
- applications
 - continuous authentication, 346
 - surveillance and re-identification, 347
- human identification using soft biometrics, 333
 - imputation, 339
- incorporating soft biometrics in fusion framework, 330
- performance metrics, 329
- predicting gender from face images, 344
- Soybean data, 106
- Sparse representation
 - robust biometrics recognition via sparse representation, 208
 - sparse representation-based classification, 206
- Spliced DNA sequence data of Genbank, 105
- Statistical methods on special manifolds for image and video understanding, 179
 - applications in image analysis
 - covariance-based detection and tracking, 196
 - 3D-model based-detection and spatio-temporal alignment, 197
 - face recognition using multiple images, 195
 - Grassmann manifold for human activity analysis, 190
 - group activity modeling using trajectories, 193
 - Hilbert sphere for modeling execution-rate variations in activities, 193
 - landmark-based facial aging and gesture analysis, 195
 - landmark-based shape analysis, 187
 - manifolds in activity analysis, 189
 - manifolds in face and gesture analysis, 195
 - manifolds in object detection, recognition, and tracking, 196
 - manifolds in shape analysis, 187
 - modeling activities as tensors, 194
 - planar curve-based shape analysis, 188
 - shape features for activity analysis, 189
 - common manifolds arising in image analysis, 185
 - differential geometric tools, 181
 - statistical analysis on manifolds, 183
 - examples, 180
- Supervised locally linear embedding (SLLE), 476
- Support Vector Machines (SVM), 254, 358
 - advantages of SVM, 361
 - for binary classification, 254
 - Linear SVM, 359
 - for multiple classes, 254
 - nonlinear SVM, 360
- SVM. *See* Support Vector Machines (SVM)

T

Text recognition, features for, 455

- chain code and critical point features, 456
 - directional profile features, 457
 - discrete symbol features, 458
 - feature selection and combination, 459
 - gradient-structural-concavity (GSC) features, 455
 - Transformed likelihood ratio (TLR), 29–30
- U**
- Undirected graphical models, 229
 - for labeling, 231
 - User behavior monitoring and profiling scheme for masquerade detection, 354
 - data collection, 361
 - data sanitization, 362
 - issues with data collection and use, 362
 - experimental design, 367
 - classification rate issues, 376
 - comprehensive data, 370
 - mouse data, 368
 - feature extraction, 363
 - design of feature extraction engine, 366
 - feature vector generation, 367
 - sliding window protocol, 365
 - feature selection methods, 363
 - Masquerade detection based on GUI data, 358
 - previous anomaly/masquerade detection efforts, 357
 - Support Vector Machines (SVM), 358
 - advantages of SVM, 361
 - Linear SVM, 359
 - nonlinear SVM, 360
- V**
- Video activity recognition by luminance differential trajectory and aligned projection distance, 302
 - Cambridge hand gesture dataset, 319
 - analysis, 321
 - simulation result, 320
 - DLFT solutions, 313
- experiments, 313
 - KTH human action dataset, 314
 - simulation result, 315
 - LAPD solutions, 313
 - problem formulation, 305
 - differential luminance field trajectory approach, 311
 - dimensionality reduction, 307
 - luminance aligned projection distance approach, 309
 - maximum likelihood detection, 308
 - video representation, 307
 - related work, 303
 - Youtube action dataset, 321
 - simulation result, 322
- Visual search**
- benchmark data sets, 289
 - data everywhere, 269
 - importance of big data, 272
 - information extraction and representation, 273
 - color, 276
 - local features, 281
 - shape, 276
 - texture, 276
 - matching images, 282
 - geometric matching, 282
 - histogram comparison, 282
 - query expansion, 285
 - memory footprint and speed, 285
 - algorithm performance, data dependency of, 285
 - approximate nearest neighbor, 287
 - compact binary code, 287
 - nearest neighbour, 286
 - outline, 270
 - scope, 270
- Y**
- Youtube action dataset, 321
 - simulation result, 322

This page is intentionally left blank