Dimitrios Gunopulos
Thomas Hofmann
Donato Malerba
Michalis Vazirgiannis (Eds.)

LNAI 6912

Machine Learning and Knowledge Discovery in Databases

European Conference, ECML PKDD 2011
Athens, Greece, September 2011
Proceedings, Part II

2 Part II

Springer

Lecture Notes in Artificial Intelligence 6912

Subseries of Lecture Notes in Computer Science

Dimitrios Gunopulos Thomas Hofmann
Donato Malerba Michalis Vazirgiannis (Eds.)

Machine Learning and Knowledge Discovery in Databases

European Conference, ECML PKDD 2011
Athens, Greece, September 5-9, 2011
Proceedings, Part II

Springer

Volume Editors

Dimitrios Gunopulos
University of Athens, Greece
E-mail: dg@di.uoa.gr

Thomas Hofmann
Google Switzerland GmbH, Zurich, Switzerland
E-mail: thofmann@google.com

Donato Malerba
University of Bari "Aldo Moro", Bari, Italy
E-mail: malerba@di.uniba.it

Michalis Vazirgiannis
Athens University of Economics and Business, Greece
E-mail: mvazirg@aueb.gr

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-23782-9 e-ISBN 978-3-642-23783-6
DOI 10.1007/978-3-642-23783-6
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.2, H.2.8, H.2, H.3, G.3, J.1, I.7, F.2.2, F.4.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Welcome to ECML PKDD 2011

Welcome to the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2011) held in Athens, Greece, during September 5–9, 2011. ECML PKDD is an annual conference that provides an international forum for the discussion of the latest high-quality research results in all areas related to machine learning and knowledge discovery in databases as well as other innovative application domains. Over the years it has evolved as one of the largest and most selective international conferences in machine learning and data mining, the only one that provides a common forum for these two closely related fields.

ECML PKDD 2011 included all the scientific events and activities of big conferences. The scientific program consisted of technical presentations of accepted papers, plenary talks by distinguished keynote speakers, workshops and tutorials, a discovery challenge track, as well as demo and industrial tracks. Moreover, two co-located workshops were organized on related research topics. We expect that all those scientific activities provide opportunities for knowledge dissemination, fruitful discussions and exchange of ideas among people both from academia and industry. Moreover, we hope that this conference will continue to offer a unique forum that stimulates and encourages close interaction among researchers working on machine learning and data mining.

We were very happy to have the conference back in Greece after 1995 when ECML was successfully organized in Heraklion, Crete. However, this was the first time that the joint ECML PKDD event was organized in Greece and, more specifically, in Athens, with the conference venue boasting a superb location under the Acropolis and in front of the Temple of Zeus. Besides the scientific activities, the conference offered delegates an attractive range of social activities, such as a welcome reception on the roof garden of the conference venue directly facing the Acropolis hill, a poster session at "Technopolis" Gazi industrial park, the conference banquet, and a farewell party at the new Acropolis Museum, one of the most impressive archaeological museums worldwide, which included a guided tour of the museum exhibits.

Several people worked hard together as a superb dedicated team to ensure the successful organization of this conference. First, we would like to express our thanks and deep gratitude to the PC Chairs Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba and Michalis Vazirgiannis. They efficiently carried out the enormous task of coordinating the rigorous hierarchical double-blind review process that resulted in a rich, while at the same time, very selective scientific program. Their contribution was crucial and essential in all phases and aspects of the conference organization and it was by no means restricted only to the paper review process. We would also like to thank the Area Chairs and Program Committee members for the valuable assistance they offered to the PC Chairs in their

timely completion of the review process under strict deadlines. Special thanks should also be given to the Workshop Co-chairs, Bart Goethals and Katharina Morik, the Tutorial Co-chairs, Fosca Giannotti and Maguelonne Teisseire, the Discovery Challenge Co-chairs, Alexandros Kalousis and Vassilis Plachouras, the Industrial Session Co-chairs, Alexandros Ntoulas and Michail Vlachos, the Demo Track Co-chairs, Michelangelo Ceci and Spiros Papadimitriou, and the Best Paper Award Co-chairs, Sunita Sarawagi and Michèle Sebag. We further thank the keynote speakers, workshop organizers, the tutorial presenters and the organizers of the discovery challenge.

Furthermore, we are indebted to the Publicity Co-chairs, Annalisa Appice and Grigorios Tsoumakas, who developed and implemented an effective dissemination plan and supported the Program Chairs in the production of the proceedings, and also to Margarita Karkali for the development, support and timely update of the conference website. We further thank the members of the ECML PKDD Steering Committee for their valuable help and guidance.

The conference was financially supported by the following generous sponsors who are worthy of special acknowledgment: Google, Pascal2 Network, Xerox, Yahoo Labs, COST-MOVE Action, Rapid-I, FP7-MODAP Project, Athena RIC / Institute for the Management of Information Systems, Hellenic Artificial Intelligence Society, Marathon Data Systems, and Transinsight. Additional support was generously provided by Sony, Springer, and the UNESCO Privacy Chair Program. This support has given us the opportunity to specify low registration rates, provide video-recording services and support students through travel grants for attending the conference. The substantial effort of the Sponsorship Co-chairs, Ina Lauth and Ioannis Kopanakis, was crucial in order to attract these sponsorships, and therefore, they deserve our special thanks. Special thanks should also be given to the five organizing institutions, namely, University of Bari "Aldo Moro", Athens University of Economics and Business, University of Athens, University of Ioannina, and University of Piraeus for supporting in multiple ways our task.

We would like to especially acknowledge the members of the Local Organization team, Maria Halkidi, Despoina Kopanaki and Nikos Pelekis, for making all necessary local arrangements and Triaena Tours & Congress S.A. for efficiently handling finance and registrations. The essential contribution of the student volunteers also deserves special acknowledgment.

Finally, we are indebted to all researchers who considered this conference as a forum for presenting and disseminating their research work, as well as to all conference participants, hoping that this event will stimulate further expansion of research and industrial activities in machine learning and data mining.

July 2011

Aristidis Likas
Yannis Theodoridis

Preface

The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2011) took place in Athens, Greece, during September 5–9, 2011. This year we have completed the first decade since the junction between the European Conference on Machine Learning and the Principles and Practice of Knowledge Discovery in Data Bases conferences, which as independent conferences date back to 1986 and 1997, respectively. During this decade there has been an increasing integration of the two fields, as reflected by the rising number of submissions of top-quality research results. In 2008 a single ECML PKDD Steering Committee was established, which gathered senior members of both communities.

The ECML PKDD conference is a highly selective conference in both areas, the leading forum where researchers in machine learning and data mining can meet, present their work, exchange ideas, gain new knowledge and perspectives, and motivate the development of new interesting research results. Although traditionally based in Europe, ECML PKDD is also a truly international conference with rich and diverse participation.

In 2011, as in previous years, ECML PKDD followed a full week schedule, from Monday to Friday. It featured six plenary invited talks by Rakesh Agrawal, Albert-László Barabási, Christofer Bishop, Andrei Broder, Marco Gori and Heikki Mannila. Monday and Friday were devoted to workshops selected by Katharina Morik and Bart Goethals, and tutorials, organized and selected by Fosca Giannotti and Maguelonne Teisseire. There was also an interesting industrial session, managed by Alexandros Ntoulas and Michalis Vlachos, which welcomed distinguished speakers from the ML and DM industry: Vasilis Aggelis, Radu Jurca, Neel Sundaresan and Olivier Verscheure. The 2011 discovery challenge was organized by Alexandros Kalousis and Vassilis Plachouras.

The main conference program unfolded from Tuesday to Thursday, where 121 papers selected among 599 full-paper submissions were presented in the technical parallel sessions and in a poster session open to all accepted papers. The acceptance rate of 20% supports the traditionally high standards of the joint conference. The selection process was assisted by 35 Area Chairs, each supervising the reviews and discussions of about 17 papers, and by 270 members of the Program Committee, with the help of 197 additional reviewers. While the selection process was made particularly intense due to the very high number of submissions, we are grateful and heartily thank all Area Chairs, members of the Program Committee, and additional reviewers for their commitment and hard work during the tight reviewing period.

The composition of the paper topics covered a wide spectrum of issues. A significant portion of the accepted papers dealt with core issues such as supervised

and unsupervised learning with some innovative contributions in fundamental issues such as cluster-ability of a dataset.

Other fundamental issues tackled by accepted papers include dimensionality reduction, distance and similarity learning, model learning and matrix/tensor analysis. In addition, there was a significant cluster of papers with valuable contributions on graph mining, graphical models, hidden Markov models, kernel methods, active and ensemble learning, semi-supervised and transductive learning, mining sparse representations, model learning, inductive logic programming, and statistical learning.

A significant part of the program covered novel and timely applications of data mining and machine learning in industrial domains, including: privacy-preserving and discrimination-aware mining, spatiotemporal data mining, text mining, topic modeling, learning from environmental and scientific data, Web mining and Web search, link analysis, bio/medical data, data Streams and sensor data, ontology-based data, relational data mining, learning from time series data, time series data.

In the past three editions of the joint conference, the two Springer journals *Machine Learning* and *Data Mining and Knowledge Discovery* published the top papers in two special issues printed in advance of the conference. These papers were not included in the conference proceedings, so there was no double publication of the same work. A novelty introduced this year was the post-conference publication of the special issues in order to guarantee the expected high-standard reviews for top-quality journals. Therefore, authors of selected machine learning and data mining papers were invited to submit a significantly extended version of their paper to the special issues. The selection was made by Program Chairs on the basis of their exceptional scientific quality and high impact on the field, as indicated by conference reviewers.

Following an earlier tradition, the Best Paper Chairs Sunita Sarawagi and Michèle Sebag contributed to the selection of papers deserving the Best Paper Awards and Best Student Paper Awards in Machine Learning and in Data Mining, sponsored by Springer. As ECML PKDD completes 10 years of joint organization, the PC chairs, together with the steering committee, initiated a 10-year Awards series. This award is established for the author(s), whose paper appeared in the ECML PKDD conference 10 years ago, and had the most impact on the machine learning and data mining research since then. This year's, first award, committee consisted of three PC co-chairs (Dimitrios Gunopulos, Donato Malerba and Michalis Vazirgiannis) and three Steering Committee members (Wray Buntine, Bart Goethals and Michèle Sebag).

The conference also featured a demo track, managed by Michelangelo Ceci and Spiros Papadimitriou; 11 demos out of 21 submitted were selected by a Demo Track Program Committee, presenting prototype systems that illustrate the high impact of machine learning and data mining application in technology. The demo descriptions are included in the proceedings. We further thank the members of the Demo Track Program Committee for their efforts in timely reviewing submitted demos.

Finally, we would like to thank the General Chairs, Aristidis Likas and Yannis Theodoridis, for their critical role in the success of the conference, the Tutorial, Workshop, Demo, Industrial Session, Discovery Challenge, Best Paper, and Local Chairs, the Area Chairs and all reviewers, for their voluntary, highly dedicated and exceptional work, and the ECML PKDD Steering Committee for their help and support. Our last and warmest thanks go to all the invited speakers, the speakers, all the attendees, and especially to the authors who chose to submit their work to the ECML PKDD conference and thus enabled us to build up this memorable scientific event.

July 2011

Dimitrios Gunopulos
Thomas Hofmann
Donato Malerba
Michalis Vazirgiannis

Organization

ECML/PKDD 2011 was organized by the Department of Computer Science—University of Ioannina, Department of Informatics—University of Piraeus, Department of Informatics and Telecommunications—University of Athens, Google Inc. (Zurich), Dipartimento di Informatica—Università degli Studi di Bari "Aldo Moro," and Department of Informatics—Athens University of Economics & Business.

General Chairs

Aristidis Likas	University of Ioannina, Greece
Yannis Theodoridis	University of Piraeus, Greece

Program Chairs

Dimitrios Gunopulos	University of Athens, Greece
Thomas Hofmann	Google Inc., Zurich, Switzerland
Donato Malerba	University of Bari "Aldo Moro," Italy
Michalis Vazirgiannis	Athens University of Economics and Business, Greece

Workshop Chairs

Katharina Morik	University of Dortmund, Germany
Bart Goethals	University of Antwerp, Belgium

Tutorial Chairs

Fosca Giannotti	Knowledge Discovery and Delivery Lab, Italy
Maguelonne Teisseire	TETIS Lab. Department of Information System and LIRMM Lab. Department of Computer Science, France

Best Paper Award Chairs

Sunita Sarawagi	Computer Science and Engineering, IIT Bombay, India
Michèle Sebag	University of Paris-Sud, France

Industrial Session Chairs

Alexandros Ntoulas	Microsoft Research, USA
Michail Vlachos	IBM Zurich Research Laboratory, Switzerland

Demo Track Chairs

Michelangelo Ceci	University of Bari "Aldo Moro," Italy
Spiros Papadimitriou	Google Research

Discovery Challenge Chairs

Alexandros Kalousis	University of Geneva, Switzerland
Vassilis Plachouras	Athens University of Economics and Business, Greece

Publicity Chairs

Annalisa Appice	University of Bari "Aldo Moro," Italy
Grigorios Tsoumakas	Aristotle University of Thessaloniki, Greece

Sponsorship Chairs

Ina Lauth	IAIS Fraunhofer, Germany
Ioannis Kopanakis	Technological Educational Institute of Crete, Greece

Organizing Committee

Maria Halkidi	University of Pireaus, Greece
Despina Kopanaki	University of Piraeus, Greece
Nikos Pelekis	University of Piraeus, Greece

Steering Committee

José Balcázar	Bart Goethals
Francesco Bonchi	Katharina Morik
Wray Buntine	Dunja Mladenic
Walter Daelemans	John Shawe-Taylor
Aristides Gionis	Michèle Sebag

Area Chairs

Elena Baralis
Hendrik Blockeel
Francesco Bonchi
Gautam Das
Janez Demsar
Amol Deshpande
Carlotta Domeniconi
Tapio Elomaa
Floriana Esposito
Fazel Famili
Wei Fan
Peter Flach
Johannes Furnkranz
Aristides Gionis
George Karypis
Ravi Kumar
James Kwok
Stan Matwin
Michael May
Taneli Mielikainen
Yücel Saygin
Arno Siebes
Jian Pei
Myra Spiliopoulou
Jie Tang
Evimaria Terzi
Bhavani M. Thuraisingham
Hannu Toivonen
Luis Torgo
Ioannis Tsamardinos
Panayiotis Tsaparas
Ioannis P. Vlahavas
Haixun Wang
Stefan Wrobel
Xindong Wu

Program Committee

Foto Afrati
Aijun An
Aris Anagnostopoulos
Gennady Andrienko
Ion Androutsopoulos
Annalisa Appice
Marta Arias
Ira Assent
Vassilis Athitsos
Martin Atzmueller
Jose Luis Balcazar
Daniel Barbara
Sugato Basu
Roberto Bayardo
Klaus Berberich
Bettina Berendt
Michele Berlingerio
Michael Berthold
Indrajit Bhattacharya
Marenglen Biba
Albert Bifet
Enrico Blanzieri
Konstantinos Blekas
Mario Boley
Zoran Bosnic
Marco Botta
Jean-Francois Boulicaut
Pavel Bradzil
Ulf Brefeld
Paula Brito
Wray Buntine
Toon Calders
Rui Camacho
Longbing Cao
Michelangelo Ceci
Tania Cerquitelli
Sharma Chakravarthy
Keith Chan
Vineet Chaoji
Keke Chen
Ling Chen
Xue-wen Chen
Weiwei Cheng
Yun Chi

Silvia Chiusano
Vassilis Christophides
Frans Coenen
James Cussens
Alfredo Cuzzocrea
Maria Damiani
Atish Das Sarma
Tijl De Bie
Jeroen De Knijf
Colin de la Higuera
Antonios Deligiannakis
Krzysztof Dembczynski
Anne Denton
Christian Desrosiers
Wei Ding
Ying Ding
Debora Donato
Kurt Driessens
Chris Drummond
Pierre Dupont
Saso Dzeroski
Tina Eliassi-Rad
Roberto Esposito
Nicola Fanizzi
Fabio Fassetti
Ad Feelders
Hakan Ferhatosmanoglou
Stefano Ferilli
Cesar Ferri
Daan Fierens
Eibe Frank
Enrique Frias-Martinez
Elisa Fromont
Efstratios Gallopoulos
Byron Gao
Jing Gao
Paolo Garza
Ricard Gavalda
Floris Geerts
Pierre Geurts
Aris Gkoulalas-Divanis
Bart Goethals
Vivekanand Gopalkrishnan
Marco Gori
Henrik Grosskreutz
Maxim Gurevich
Maria Halkidi
Mohammad Hasan
Jose Hernandez-Orallo
Eyke Hüllermeier
Vasant Honavar
Andreas Hotho
Xiaohua Hu
Ming Hua
Minlie Huang
Marcus Hutter
Nathalie Japkowicz
Szymon Jaroszewicz
Daxin Jiang
Alipio Jorge
Theodore Kalamboukis
Alexandros Kalousis
Panagiotis Karras
Samuel Kaski
Ioannis Katakis
John Keane
Kristian Kersting
Latifur Khan
Joost Kok
Christian Konig
Irena Koprinska
Walter Kosters
Georgia Koutrika
Stefan Kramer
Raghuram Krishnapuram
Marzena Kryszkiewicz
Nicolas Lachiche
Nada Lăvrac
Wang-Chien Lee
Feifei Li
Jiuyong Li
Juanzi Li
Tao Li
Chih-Jen Lin
Hsuan-Tien Lin
Jessica Lin
Shou-de Lin
Song Lin
Helger Lipmaa
Bing Liu

Huan Liu
Yan Liu
Corrado Loglisci
Chang-Tien Lu
Ping Luo
Panagis Magdalinos
Giuseppe Manco
Yannis Manolopoulos
Simone Marinai
Dimitrios Mavroeidis
Ernestina Menasalvas
Rosa Meo
Pauli Miettinen
Dunja Mladenic
Marie-Francine Moens
Katharina Morik
Mirco Nanni
Alexandros Nanopoulos
Benjamin Nguyen
Frank Nielsen
Siegfried Nijssen
Richard Nock
Kjetil Norvag
Irene Ntoutsi
Salvatore Orlando
Gerhard Paass
George Paliouras
Apostolos Papadopoulos
Panagiotis Papapetrou
Stelios Paparizos
Dimitris Pappas
Ioannis Partalas
Srinivasan Parthasarathy
Andrea Passerini
Vladimir Pavlovic
Dino Pedreschi
Nikos Pelekis
Jing Peng
Ruggero Pensa
Bernhard Pfahringer
Fabio Pinelli
Enric Plaza
George Potamias
Michalis Potamias
Doina Precup
Kunal Punera
Chedy Raissi
Jan Ramon
Huzefa Rangwala
Zbigniew Ras
Ann Ratanamahatana
Jan Rauch
Matthias Renz
Christophe Rigotti
Fabrizio Riguzzi
Celine Robardet
Marko Robnik-Sikonja
Pedro Rodrigues
Fabrice Rossi
Juho Rousu
Celine Rouveirol
Ulrich Rückert
Salvatore Ruggieri
Stefan Ruping
Lorenza Saitta
Ansaf Salleb-Aouissi
Claudio Sartori
Lars Schmidt-Thieme
Matthias Schubert
Michèle Sebag
Thomas Seidl
Prithviraj Sen
Andrzej Skowron
Carlos Soares
Yangqiu Song
Alessandro Sperduti
Jerzy Stefanowski
Jean-Marc Steyaert
Alberto Suarez
Johan Suykens
Einoshin Suzuki
Panagiotis Symeonidis
Marcin Szczuka
Andrea Tagarelli
Domenico Talia
Pang-Ning Tan
Letizia Tanca
Lei Tang
Dacheng Tao
Nikolaj Tatti

Martin Theobald
Dimitrios Thilikos
Jilei Tian
Ivor Tsang
Grigorios Tsoumakas
Theodoros Tzouramanis
Antti Ukkonen
Takeaki Uno
Athina Vakali
Giorgio Valentini
Maarten van Someren
Iraklis Varlamis
Julien Velcin
Celine Vens
Jean-Philippe Vert
Vassilios Verykios
Herna Viktor
Jilles Vreeken
Willem Waegeman
Jianyong Wang
Wei Wang
Xuanhui Wang
Hui Xiong
Jieping Ye
Jeffrey Yu
Philip Yu
Bianca Zadrozny
Gerson Zaverucha
Demetris Zeinalipour
Filip Zelezny
Changshui Zhang
Kai Zhang
Kun Zhang
Min-Ling Zhang
Nan Zhang
Shichao Zhang
Zhongfei Zhang
Junping Zhang
Ying Zhao
Bin Zhou
Zhi-Hua Zhou
Kenny Zhu
Xingquan Zhu
Djamel Zighed
Indre Zliobaite
Blaz Zupan

Additional Reviewers

Pedro Abreu
Raman Adaikkalavan
Artur Aiguzhinov
Darko Aleksovski
Tristan Allard
Alessia Amelio
Panayiotis Andreou
Fabrizio Angiulli
Josephine Antoniou
Andrea Argentini
Krisztian Balog
Teresa M.A. Basile
Christian Beecks
Antonio Bella
Aurelien Bellet
Dominik Benz
Thomas Bernecker
Alberto Bertoni
Jerzy Blaszczynski
Brigitte Boden
Samuel Branders
Janez Brank
Agnès Braud
Giulia Bruno
Luca Cagliero
Yuanzhe Cai
Ercan Canhas
Eugenio Cesario
George Chatzimilioudis
Xi Chen
Anna Ciampi
Marek Ciglan
Tyler Clemons
Joseph Cohen
Carmela Comito
Andreas Constantinides
Michael Corsello
Michele Coscia

Gianni Costa
Claudia D'Amato
Mayank Daswani
Gerben de Vries
Nicoletta Del Buono
Elnaz Delpisheh
Elnaz Delpisheh
Engin Demir
Nicola Di Mauro
Ivica Dimitrovski
Martin Dimkovski
Huyen Do
Lucas Drumond
Ana Luisa Duboc
Tobias Emrich
Saeed Faisal
Zheng Fang
Wei Feng
Cèsar Ferri
Alessandro Fiori
Nuno A. Fonseca
Marco Frasca
Christoph Freudenthaler
Sergej Fries
Natalja Friesen
David Fuhry
Dimitrios Galanis
Zeno Gantner
Bo Gao
Juan Carlos Gomez
Alberto Grand
Francesco Gullo
Tias Guns
Marwan Hassani
Zhouzhou He
Daniel Hernàndez-Lobato
Tomas Horvath
Dino Ienco
Elena Ikonomovska
Anca Ivanescu
Francois Jacquenet
Baptiste Jeudy
Dustin Jiang
Lili Jiang
Maria Jose
Faisal Kamiran
Michael Kamp
Mehdi Kargar
Mehdi Kaytoue
Jyrki Kivinen
Dragi Kocev
Domen Košir
Iordanis Koutsopoulos
Hardy Kremer
Anastasia Krithara
Artus Krohn-Grimberghe
Onur Kucuktunc
Tor Lattimore
Florian Lemmerich
Jun Li
Rong-Hua Li
Yingming Li
Siyi Liu
Stefano Lodi
Claudio Lucchese
Gjorgji Madjarov
M. M. Hassan Mahmud
Fernando Martínez-Plumed
Elio Masciari
Michael Mathioudakis
Ida Mele
Corrado Mencar
Glauber Menezes
Pasquale Minervini
Ieva Mitasiunaite-Besson
Folke Mitzlaff
Anna Monreale
Gianluca Moro
Alessandro Moschitti
Yang Mu
Ricardo Ñanculef
Franco Maria Nardini
Robert Neumayer
Phuong Nguyen
Stefan Novak
Tim O'Keefe
Riccardo Ortale
Aline Paes
George Pallis
Luca Pappalardo

Jérôme Paul
Daniel Paurat
Sergios Petridis
Darko Pevec
Jean-Philippe Peyrache
Anja Pilz
Marc Plantevit
Matija Polajnar
Giovanni Ponti
Adriana Prado
Xu Pu
ZhongAng Qi
Ariadna Quattoni
Alessandra Raffaetà
Maria Jose Ramírez-Quintana
Hongda Ren
Salvatore Rinzivillo
Ettore Ritacco
Matthew Robards
Christophe Rodrigues
Giulio Rossetti
André Rossi
Cláudio Sá
Venu Satuluri
Leander Schietgat
Marijn Schraagen
Wen Shao
Wei Shen
A. Pedro Duarte Silva
Claudio Silvestri
Ivica Slavkov
Henry Soldano
Yi Song
Miha Stajdohar
Anže Staric
Erik Strumbelj
Ilija Subasic
Peter Sunehag
Izabela Szczech
Frank Takes
Aditya Telang
Eleftherios Tiakas
Gabriele Tolomei
Marko Toplak
Daniel Trabold
Roberto Trasarti
Paolo Trunfio
George Tsatsaronis
Guy Van den Broeck
Antoine Veillard
Mathias Verbeke
Jan Verwaeren
Alessia Visconti
Dimitrios Vogiatzis
Dawei Wang
Jun Wang
Louis Wehenkel
Adam Woznica
Ming Yang
Qingyan Yang
Xintian Yang
Lan Zagar
Jure Žbontar
Bernard Zenko
Pin Zhao
Yuanyuan Zhu
Albrecht Zimmermann
Andreas Züfle

Sponsoring Institutions

We wish to express our gratitude to the sponsors of ECML PKDD 2011 for their essential contribution to the conference: Google, Pascal2 Network, Xerox, Yahoo Lab, COST/MOVE (Knowledge Discovery from Movin, Objects), FP7/MODAP (Mobility, Data Mining, and Privacy), Rapid-I (report the future), Athena/IMIS (Institute for the Management of Information Systems), EETN (Hellenic Artificial Intelligence Society), MDS Marathon Data Systems, Transinsight, SONY, UNESCO Privacy Chair, Springer, Università degli Studi di Bari "Aldo Moro," Athens University of Economics and Business, University of Ioannina, National and Kapodistrian University of Athens, and the University of Piraeus.

Table of Contents – Part II

Regular Papers

Common Substructure Learning of Multiple Graphical Gaussian Models

Satoshi Hara and Takashi Washio

The Institute of Scientific and Industrial Research (ISIR), Osaka University, Japan
{hara,washio}@ar.sanken.osaka-u.ac.jp

Abstract. Learning underlying mechanisms of data generation is of great interest in the scientific and engineering fields amongst others. Finding dependency structures among variables in the data is one possible approach for the purpose, and is an important task in data mining. In this paper, we focus on learning dependency substructures shared by multiple datasets. In many scenarios, the nature of data varies due to a change in the surrounding conditions or non-stationary mechanisms over the multiple datasets. However, we can also assume that the change occurs only partially and some relations between variables remain unchanged. Moreover, we can expect that such commonness over the multiple datasets is closely related to the invariance of the underlying mechanism. For example, errors in engineering systems are usually caused by faults in the sub-systems with the other parts remaining healthy. In such situations, though anomalies are observed in sensor values, the underlying invariance of the healthy sub-systems is still captured by some steady dependency structures before and after the onset of the error. We propose a structure learning algorithm to find such invariances in the case of Graphical Gaussian Models (GGM). The proposed method is based on a block coordinate descent optimization, where subproblems can be solved efficiently by existing algorithms for *Lasso* and the *continuous quadratic knapsack problem*. We confirm the validity of our approach through numerical simulations and also in applications with real world datasets extracted from the analysis of city-cycle fuel consumption and anomaly detection in car sensors.

Keywords: Graphical Gaussian Model, common substructure, block coordinate descent.

1 Introduction

In the real world, it is common that multivariate data, such as the stock market [1], gene regulatory networks [2], or biomedical measurements [3], to have a complex dependency structure among variables. Such a structure is closely tied to the intrinsic data generating mechanism, which one aims to reveal. For example, we can expect the interaction of brain sub-regions to be reflected by the dependency structures between fMRI signals [3].

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 1–16, 2011.

The dependency structure among variables also plays an important role in the analysis of multiple datasets. It is frequently seen that datasets collected under different conditions have different dependency structures, which is caused by a change in the underlying mechanism [2,4]. On the other hand, if some relations are common to several conditions, we can expect that background mechanism to have a certain invariance against the change. An illustrative example is an engineering system where system errors are observed as dependency anomalies of sensor values [5]. These are usually caused by a fault in a sub-system. The invariance, which in this example is the remaining healthy sub-systems, is captured by a steady dependency over the multiple datasets sampled before and after the error onset.

Motivated by the example above, we propose a method for finding common dependency structures from multiple datasets. In this paper, we consider the case of the Graphical Gaussian Model (GGM) [6]. GGM is one of the most basic models representing linear dependencies among continuous variables. Identification of the structure was firstly studied by Dempster [7] where it was referred to as *covariance selection*. Though classical approaches have encountered several difficulties, there is a recent development on the use of ℓ_1-regularization [8,9,10], that enables the design of an efficient Graphical Lasso (GLasso) algorithm [11]. Since this breakthrough, several extensions have been proposed [3,12,13,14,15]. For example, Zhang et al. [12] used a Fused Lasso type formulation [16] to extract structural changes in a two-sample situation. In the multi-task learning literature [17], joint estimation algorithms for a set of GGMs with the same topological structures [3,13,14] have been studied based on a group-Lasso [18], while Guo et al. [15] proposed iterative re-weighting of GLasso for estimating multiple GGMs.

Though several GGM learning methods have been proposed, to the best of our knowledge, there are no general techniques for finding a common substructure from multiple datasets[1]. In many practical situations, such as sensor data, the data is highly noisy and the estimated structures tend to have a high variance, which masks the invariance we wish to detect. The scarcity of available data is also a crucial factor in this problem. In our work, we penalized the variation in resulting structures and formulated the common substructure learning problem as an extension of the two approaches presented in Zhang et al. [12] and Honorio et al. [13], respectively. The problem is convex and the solution is obtained by adopting a block coordinate descent procedure [19]. We further show that the solution to the subproblem in the coordinate descent can be classified into three types each of which can be derived efficiently using existing methods. We confirm the validity of our approach through numerical simulations and also in an application with real world datasets derived from the analysis of city-cycle fuel consumption and anomaly detection in car sensors.

[1] Zhang et al. [12] considered a similar problem and although their approach provides a common substructure, it is limited to only two-sample situations. The approach by Chiquet et al. [14] adopted commonness only with respect to its signs.

The remainder of this paper is organized as follows; We first review existing methods for GGM learning and its extensions to joint estimation settings in Section 2. We formulate the common substructure learning problem in Section 3 and then, in Section 4, we present the block coordinate descent algorithm. Section 5 contains numerical simulations to show the validity of the proposed method using synthetic and real world data. Finally, we conclude the paper in Section 6.

2 Structure Learning of Graphical Gaussian Model

In this section, we review the GGM estimation problem [8,9,10,11] and extensions to joint estimation of multiple GGMs [12,13].

2.1 Graphical Gaussian Model

In multivariate analysis, covariance or correlation are commonly used as an indicator of the relationship between two variables. However, in general, the covariance between two variables x_j and $x_{j'}$ is affected by a third variable. Therefore, we need to remove such effects to estimate the essential dependency structure, which is obtained by searching conditional dependency among variables. If a d-dimensional random variable $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)^\top$ is Gaussian, the conditional dependency between two variables is expressed by a precision matrix $\Lambda \in \mathbb{R}^{d\times d}$ (or inverse covariance). Under multivariate Gaussian distribution, the following property exists:

$$\Lambda_{jj'} = 0 \;\Leftrightarrow\; x_j \perp\!\!\!\perp x_{j'} \mid \text{other variables} \tag{1}$$

where $\perp\!\!\!\perp$ denotes statistical independence. With this property, GGM is defined as a graph where each node corresponds to a random variable x_j and the adjacency matrix is given by Λ. In a GGM, there is an edge between two nodes only if the corresponding two variables are conditionally dependent. In the case that only a few pairs of variables are dependent, most of the off-diagonal elements in Λ are zero and the corresponding graph expression is sparse, which allows us to visually inspect the underlying relations.

2.2 Sparse Estimation of GGM

The maximum likelihood estimator of a precision matrix is given as the inverse of the sample covariance matrix $\hat{\Sigma}$. This estimator is usually dense and the corresponding GGM is a complete graph, which states that every pair of variables is dependent. The difficulty arises here in that this occurs even when the true precision matrix is sparse and masks the underlying intrinsic relationships. To avoid this unfavorable property, Meinshausen and Bühlmann [8] proposed the use of Lasso for the sparse graph identification, which is later reformulated as a ℓ_1-regularized maximum likelihood problem [9,10]:

$$\begin{aligned} &\max_{\Lambda\in\mathbb{R}^{d\times d}} \; \ell(\Lambda; \hat{\Sigma}) - \rho\|\Lambda\|_1 \\ &\text{subject to } \; \Lambda \succ 0 \end{aligned} \tag{2}$$

where ρ is a regularization parameter and $\ell(\Lambda; \hat{\Sigma})$ is the log-likelihood of a Gaussian distribution defined as

$$\ell(\Lambda; \hat{\Sigma}) = \log\det\Lambda - \mathrm{tr}\left(\hat{\Sigma}\Lambda\right) \ . \tag{3}$$

The constraint is imposed since Λ must be positive definite as a valid precision matrix. The solution to (2) is sparse due to the effect of an additional ℓ_1-regularization term. An efficient algorithm, using a block coordinate descent [11], is available to solve this problem.

2.3 Learning Structural Changes

When comparing two GGMs representing similar models, some common edges may exist whose weights are close to one another. Zhang et al. [12] proposed the use of a Fused Lasso type regularization [16] to round these similar values to exactly the same value, thus allowing only the significant differences between two GGMs to be extracted. Their original idea is based on the work of Meinshausen and Bühlmann [8], which can naturally be transformed to an ℓ_1-regularized maximum likelihood type setting:

$$\begin{aligned} \max_{\Lambda_1,\Lambda_2} \sum_{i=1}^{2} \left\{ \ell(\Lambda_i; \hat{\Sigma}_i) - \rho\|\Lambda_i\|_1 \right\} - \gamma \sum_{j\neq j'} |\Lambda_{1,jj'} - \Lambda_{2,jj'}| \\ \text{subject to } \ \Lambda_1, \Lambda_2 \succ 0 \end{aligned} \tag{4}$$

where ρ and γ are regularization parameters. The last term forces the difference between certain elements of two matrices to be zero. They also provided an efficient technique for solving the subproblem of (4) which makes the entire procedure fast.

2.4 Multi-task Approach for Learning a Set of GGMs

The ordinary GGM estimation problem (2) aims to learn a single GGM from one dataset. Apart from the GGM estimation, it is known that jointly solving multiple similar tasks often improves the learning performance, which is referred to as multi-task learning [17]. Honorio et al. [13] assumed that all GGMs have the same topological structures, i.e., the same zero patterns in all precision matrices, and adopted the group-Lasso [18] approach, which is formulated as:

$$\begin{aligned} \max_{\{\Lambda_i\}_{i=1}^N} \sum_{i=1}^{N} t_i \ell(\Lambda_i; \hat{\Sigma}_i) - \rho \sum_{j\neq j'} \max_i |\Lambda_{i,jj'}| \\ \text{subject to } \ \Lambda_1, \Lambda_2, \ldots, \Lambda_N \succ 0 \end{aligned} \tag{5}$$

where ρ is a regularization parameter and $t_1, t_2, \ldots, t_N$ are non-negative constants. The regularization term ensures that the joint structure $\tilde{\Lambda}_{jj'} = \max_i |\Lambda_{i,jj'}|$ is sparse, where $\tilde{\Lambda}_{jj'} = 0$ denotes that the corresponding (j, j')-th entries are commonly zero in all N precision matrices.

3 Common Substructure Learning

The aim of the common substructure learning is to find a dependency structure between variables that is invariant to changes in the surrounding conditions. Formally, we have N covariance matrices $\hat{\Sigma}_1, \hat{\Sigma}_2, \ldots, \hat{\Sigma}_N$ each of which is calculated from datasets sampled under different conditions. The task is to identify common elements shared by all precision matrices $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$. To begin with, we assume that the number of variables in each condition is the same, i.e., all have d-dimensions. Also, the identities of each variable are the same, e.g., x_1 is always the value from the same sensor while surrounding conditions may change. Then, we define a common substructure of multiple GGMs as follows:

Definition 1 (Common Substructure of Multiple GGMs) *Let $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$ be the corresponding precision matrices of each GGM. Then, their common substructure is expressed by an adjacency matrix Θ defined as*

$$\Theta_{jj'} = \begin{cases} \Lambda_{1,jj'}\ , & \text{if } \Lambda_{1,jj'} = \Lambda_{2,jj'} = \ldots = \Lambda_{N,jj'} \\ 0\ , & \text{otherwise} \end{cases} . \tag{6}$$

The common substructure defined here has an edge between nodes only if the corresponding edge weights among all GGMs are equal. We expect to find such a substructure in the estimated precision matrices. To that end, we impose two regularizations and formulate the following problem:

$$\begin{aligned} &\max_{\{\Lambda_i\}_{i=1}^N} \sum_{i=1}^{N} t_i\, \ell(\Lambda_i; \hat{\Sigma}_i) - \sum_{j \neq j'} \left(\rho \max_i |\Lambda_{i,jj'}| + \gamma \max_{i,i'} |\Lambda_{i,jj'} - \Lambda_{i',jj'}| \right) \\ &\quad \text{subject to } \ \Lambda_1,\ \Lambda_2,\ \ldots,\ \Lambda_N \succ 0 \end{aligned} \tag{7}$$

where ρ, γ are regularization parameters and $t_1, t_2, \ldots, t_N$ are non-negative constants that satisfy $\sum_{i=1}^N t_i = 1$. Here, constants t_i are weighting parameters, usually chosen as $t_i = n_i / \sum_{i=1}^N n_i$ where n_i is the size of the i-th dataset. The second regularization term is a generalization of the one in (4) for $N \geq 3$, which ensures that some entries in the resulting precision matrices are common to all matrices. Since the second regularization does not impose any sparsity on the resulting precision matrices, we added the joint regularization term appearing in (5). The resulting common substructure Θ is obtained by applying Definition 1 to the estimated precision matrices $\hat{\Lambda}_1, \hat{\Lambda}_2, \ldots, \hat{\Lambda}_N$.

4 Algorithm

The problem (7) is a concave maximization with convex constraints. In this section, we introduce the solution algorithm based on the block coordinate descent method [19], where the approach is justified by the following theorem.

Theorem 1. *The solution sequence generated by the block coordinate descent for problem (7) is bounded and every cluster point*[2] *is a solution.*

[2] A point where the sequence converges.

4.1 Block Coordinate Descent

In the block coordinate descent, we fix elements in Λ_i corresponding to variables $x_1, \ldots, x_{m-1}, x_{m+1}, \ldots, x_d$ and update entries related to a variable x_m. Since (7) is invariant for permutations of rows and columns in matrices, we can always arrange x_m-related entries located in the last row and column. Then, we partition each matrix into four parts, namely, one matrix, two vectors, and a scalar:

$$\Lambda_i = \begin{bmatrix} Z_i & \boldsymbol{z}_i \\ \boldsymbol{z}_i^\top & \omega_i \end{bmatrix} , \quad \hat{\Sigma}_i = \begin{bmatrix} P_i & \boldsymbol{p}_i \\ \boldsymbol{p}_i^\top & q_i \end{bmatrix} . \tag{8}$$

Now, we fix $Z_1, Z_2, \ldots, Z_N$ and derive the subproblem on $\{\boldsymbol{z}_i, \omega_i\}_{i=1}^N$:

$$\max_{\{\boldsymbol{z}_i, \omega_i\}_{i=1}^N} \sum_{i=1}^N t_i \left\{ \log\left(\omega_i - \boldsymbol{z}_i^\top Z_i^{-1} \boldsymbol{z}_i\right) - 2\boldsymbol{p}_i^\top \boldsymbol{z}_i - q_i \omega_i \right\} \\ -2 \sum_j \left(\rho \max_i |z_{ij}| + \gamma \max_{i,i'} |z_{ij} - z_{i'j}| \right) \tag{9}$$

where z_{ij} is the j-th entry of $\boldsymbol{z}_i$. By setting the derivative over ω_i to zero, we get:

$$\omega_i = \boldsymbol{z}_i^\top Z_i^{-1} \boldsymbol{z}_i + q_i^{-1} . \tag{10}$$

Here, $Z_i \succ 0$ and $\omega_i - \boldsymbol{z}_i^\top Z_i^{-1} \boldsymbol{z}_i = q_i^{-1} > 0$ guarantee the positive definiteness of Λ_i. Therefore, by choosing the initial Λ_i to be positive definite, that property is always preserved by the updating procedure of the block coordinate descent. Next, by substituting (10) into (9), we derive:

$$\min_{\{\boldsymbol{z}_i\}_{i=1}^N} \sum_{i=1}^N t_i \left(\frac{q_i}{2} \boldsymbol{z}_i^\top Z_i^{-1} \boldsymbol{z}_i + \boldsymbol{p}_i^\top \boldsymbol{z}_i \right) + \sum_j \left(\rho \max_i |z_{ij}| + \gamma \max_{i,i'} |z_{ij} - z_{i'j}| \right) . \tag{11}$$

Instead of solving this problem, we again adopt a coordinate descent approach and further decompose it into subproblems. We solve (11) only for elements related to the variable $x_{m'}$ $(m' \neq m)$ and fix the other entries. As before, we arrange the corresponding elements into the last of the vectors and matrices:

$$\boldsymbol{z}_i = \begin{bmatrix} \boldsymbol{v}_i \\ w_i \end{bmatrix} , \quad \boldsymbol{p}_i = \begin{bmatrix} \boldsymbol{r}_i \\ s_i \end{bmatrix} , \quad \tilde{Z}_i^{-1} = \begin{bmatrix} H_i & \boldsymbol{h}_i \\ \boldsymbol{h}_i^\top & g_i \end{bmatrix} . \tag{12}$$

Then, we derive the following subproblem of (11) over $\boldsymbol{w} = (w_1, w_2, \ldots, w_N)^\top$:

$$\min_{\boldsymbol{w}} \ \frac{1}{2} \boldsymbol{w}^\top \mathrm{diag}(\boldsymbol{a}) \boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w} + \rho \|\boldsymbol{w}\|_\infty + \gamma \max_{i,i'} |w_i - w_{i'}| \tag{13}$$

with coefficients $a_i = t_i q_i g_i$ and $b_i = -t_i \left(q_i \boldsymbol{h}_i^\top \boldsymbol{v}_i + s_i \right)$. The dual problem is

$$\min_{\boldsymbol{\xi}} \frac{1}{2} (\boldsymbol{b} - \boldsymbol{\xi})^\top \mathrm{diag}(\boldsymbol{a})^{-1} (\boldsymbol{b} - \boldsymbol{\xi}) \\ \text{subject to } \ |\boldsymbol{1}_N^\top \boldsymbol{\xi}| \leq \rho, \ \ \|\boldsymbol{\xi}\|_1 \leq \rho + 2\gamma \tag{14}$$

where $\boldsymbol{\xi} = \boldsymbol{b} - \mathrm{diag}(\boldsymbol{a}) \boldsymbol{w}$. This is the subproblem for the block coordinate descent of (7). In the next section, we show that this problem has three types of solutions each which can be derived efficiently.

4.2 Subproblem

First, we can see that subproblem (14) has a solution $\boldsymbol{\xi} = \boldsymbol{b}$ when $|\mathbf{1}_N^\top \boldsymbol{b}| \leq \rho$ and $\|\boldsymbol{b}\|_1 \leq \rho + 2\gamma$. In the case of $|\mathbf{1}_N^\top \boldsymbol{b}| > \rho$ or $\|\boldsymbol{b}\|_1 > \rho + 2\gamma$, the solution is on the boundary of the constraint set and can be classified into three types. Here, we give the solution procedure for each of these. The entire procedure is summarized in Algorithm 1.

1) The solution is on the boundary $\|\boldsymbol{\xi}\|_1 = \rho + 2\gamma$: In this case, we ignore the first constraint in (14) and solve only for the second constraint. Moreover, this problem is shown to be equivalent to the following *continuous quadratic knapsack problem* [13]:

$$\min_{\boldsymbol{y}} \sum_{i=1}^{N} \frac{1}{2a_i} \left(|b_i| - y_i\right)^2 \quad \text{subject to} \quad \boldsymbol{y} \geq 0\,, \quad \mathbf{1}_N^\top \boldsymbol{y} = \rho + 2\gamma \tag{15}$$

which relates to $\boldsymbol{\xi}$ by $\xi_i = \mathrm{sgn}(b_i) y_i$ where $\mathrm{sgn}(*)$ is a sign function. We give the solution procedure for this problem [13] in Section 4.3. Here, we note that the resulting $\boldsymbol{\xi}$ may violate the constraint $|\mathbf{1}_N^\top \boldsymbol{\xi}| \leq \rho$ since we have ignored it. In this case, we discard the solution and move on to the next case.

2) The solution is on the boundary $|\mathbf{1}_N^\top \boldsymbol{\xi}| = \rho$: This time, we ignore the second constraint in (14) and solve

$$\min_{\boldsymbol{\xi}} \frac{1}{2} (\boldsymbol{b} - \boldsymbol{\xi})^\top \mathrm{diag}(\boldsymbol{a})^{-1} (\boldsymbol{b} - \boldsymbol{\xi}) \quad \text{subject to} \quad |\mathbf{1}_N^\top \boldsymbol{\xi}| \leq \rho\,. \tag{16}$$

This problem has the following single variable Lasso for its dual:

$$\min_{w_0} \frac{a}{2} w_0^2 - b w_0 + \rho |w_0| \tag{17}$$

with $a = \sum_{i=1}^N a_i$ and $b = \sum_{i=1}^N b_i$, and the solution is obtained as

$$w_0 = \mathrm{sgn}(b) \frac{(|b| - \rho)_+}{a} \tag{18}$$

where $(*)_+ = \max(*, 0)$ is a soft-thresholding operator. Again, the resulting value $\boldsymbol{\xi} = \boldsymbol{b} - w_0 \boldsymbol{a}$ may violate $\|\boldsymbol{\xi}\|_1 \leq \rho + 2\gamma$. In this case, the solution is on the edge of the intersection of two constraints, and is obtained by the next procedure.

3) The solution is on both boundaries $|\mathbf{1}_N^\top \boldsymbol{\xi}| = \rho$ and $\|\boldsymbol{\xi}\|_1 = \rho + 2\gamma$: Here, we solve (14) with two equality constraints. The procedure in this section is based on the following theorem.

Theorem 2. *Let the solution to (16) be $\tilde{\boldsymbol{\xi}}$. Then, the solution to (14) has the same signs as $\tilde{\boldsymbol{\xi}}$, i.e. $\tilde{\xi}_i \xi_i \geq 0$ for $1 \leq i \leq N$.*

From this result, we can factorize the objective function into the sum of two components $\sum_{\tilde{\xi}_i \geq 0} \frac{1}{2a_i}(b_i - \xi_i)^2$ and $\sum_{\tilde{\xi}_i < 0} \frac{1}{2a_i}(b_i - \xi_i)^2$. The constraint terms can also be expressed as $\sum_{\tilde{\xi}_i \geq 0} \xi_i + \sum_{\tilde{\xi}_i < 0} \xi_i = \rho$ (or $-\rho$) and $\sum_{\tilde{\xi}_i \geq 0} \xi_i - \sum_{\tilde{\xi}_i < 0} \xi_i = \rho + 2\gamma$. As a result, we derive two independent problems:

$$\min_{\boldsymbol{y}^+} \sum_{\tilde{\xi}_i \geq 0} \frac{1}{2a_i} \left(y_i^+ - b_i\right)^2 \quad \text{subject to} \quad \boldsymbol{y}^+ \geq 0 \,, \quad \sum_{\tilde{\xi}_i \geq 0} y_i^+ = \alpha^+ \,, \tag{19}$$

$$\min_{\boldsymbol{y}^-} \sum_{\tilde{\xi}_i < 0} \frac{1}{2a_i} \left(y_i^- + b_i\right)^2 \quad \text{subject to} \quad \boldsymbol{y}^- \geq 0 \,, \quad \sum_{\tilde{\xi}_i < 0} y_i^- = \alpha^- \,. \tag{20}$$

The solutions to these problems relate to $\boldsymbol{\xi}$ in that $\xi_i = y_i^+$ for $\tilde{\xi}_i \geq 0$ and $\xi_i = -y_i^-$ for $\tilde{\xi}_i < 0$. The parameters α^+ and α^- are $\rho+\gamma$ and γ, respectively if the solution is on $\mathbf{1}_N^\top \boldsymbol{\xi} = \rho$, and γ and $\rho+\gamma$, respectively, for $\mathbf{1}_N^\top \boldsymbol{\xi} = -\rho$. These problems are once again continuous quadratic knapsack problems and the solutions can be efficiently obtained by using the algorithm presented in [13]. We can derive the final solution by solving these problems for both cases $\mathbf{1}_N^\top \boldsymbol{\xi} = \rho$ and $\mathbf{1}_N^\top \boldsymbol{\xi} = -\rho$, and choosing the one with the smaller objective function value in (14).

4.3 Continuous Quadratic Knapsack Problem

In this section, we briefly summarize the algorithm for solving the following continuous quadratic knapsack problem presented in [13]:

$$\min_{\boldsymbol{y}} \sum_{i=1}^{N} \frac{1}{2c_i} \left(y_i - d_i\right)^2 \quad \text{subject to} \quad \boldsymbol{y} \geq 0 \,, \quad \mathbf{1}_N^\top \boldsymbol{y} = \alpha \,. \tag{21}$$

Note that this formulation is common to (15), (19) and (20). From the KKT condition, the solution to this problem is given as $y_i(\nu) = \max(d_i - \nu c_i, 0)$ with some constant ν. Moreover, the optimal ν is what satisfies $\mathbf{1}_N^\top \boldsymbol{y}(\nu) = \alpha$. Since $\mathbf{1}_N^\top \boldsymbol{y}(\nu)$ is a decreasing piecewise linear function with breakpoints $\frac{d_i}{c_i}$, we can find a minimum breakpoint $\nu_0 = \frac{d_{i_0}}{c_{i_0}}$ that satisfies $\mathbf{1}_N^\top \boldsymbol{y}(\nu_0) \leq \alpha$ by sorting the N breakpoints. Then, the optimal ν is given as

$$\nu = \frac{\sum_{d_i - \nu_0 c_i \geq 0} d_i - \alpha}{\sum_{d_i - \nu_0 c_i \geq 0} c_i} \,. \tag{22}$$

4.4 Hyper-Parameters ρ and γ

The choice of hyper-parameters ρ and γ affects the resulting graphical models. There are several approaches for choosing these, such as cross validation [9,15] or the Bayesian information criterion [15]. Apart from selection techniques, the following result gives us some insight into ρ and γ, and is helpful for analyzing the data more intensively.

Algorithm 1. Pseudo Code for Common Substructure Learning

Input : sample covariances $\hat{\Sigma}_1, \hat{\Sigma}_2, \ldots, \hat{\Sigma}_N$, regularization parameters ρ, γ
constants $t_1, t_2, \ldots, t_N > 0$, $\sum_{i=1}^N t_i = 1$
Output : precision matrices $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$

initialize $\Lambda_i \leftarrow \hat{\Sigma}_i^{-1}$ for each $1 \leq i \leq N$;
repeat
  for $x_m : m = 1$ to d **do**
    for $x_{m'} : m' \neq m$ **do**
      if $|\mathbf{1}_N^\top \boldsymbol{b}| \leq \rho$ and $\|\boldsymbol{b}\|_1 \leq \rho + 2\gamma$ **then**
        $\boldsymbol{\xi} \leftarrow \boldsymbol{b}$;
      else
        solve continuous quadratic knapsack problem (15);
        if the solution does not satisfy $|\mathbf{1}_N^\top \boldsymbol{\xi}| \leq \rho$ **then**
          solve (16) with single variable Lasso;
          if the solution does not satisfy $\|\boldsymbol{\xi}\|_1 \leq \rho + 2\gamma$ **then**
            solve (19) and (20) for $[\alpha^+, \alpha^-] = [\rho + \gamma, \gamma]$;
            solve (19) and (20) for $[\alpha^+, \alpha^-] = [\gamma, \rho + \gamma]$;
            adopt one of the two solutions with the smaller value for (14);
          end if
        end if
      end if
      $\boldsymbol{w} \leftarrow \mathrm{diag}(\boldsymbol{a})^{-1}(\boldsymbol{b} - \boldsymbol{\xi})$;
      update (m, m')-th and (m', m)-th elements of Λ_i with w_i for $1 \leq i \leq N$;
    end for
    update (m, m)-th element of Λ_i by (10);
  end for
until $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$ converges

Proposition 1. *In the bivariate case, the off-diagonal elements of the precision matrices λ_i have the following property:*

$$|r_i| \leq \rho + 2\gamma \;\; \textit{for} \;\; 1 \leq i \leq N \;\; \textit{and} \;\; \left| \sum_{i=1}^N t_i r_i \right| \leq \rho \;\; \Rightarrow \;\; \lambda_i = 0 \tag{23}$$

where r_i is the covariance between two variables in the i-th dataset.

Although the result is specific to the bivariate case, we can interpret $\tilde{\gamma} = \rho + 2\gamma$ and ρ as thresholding parameters. If we wish to treat dependencies higher than some level as significant and expect them to be non-zero, $\tilde{\gamma}$ should not exceed that level. We can also see that ρ is the threshold for the average covariance and the parameter that controls the existence of common substructures.

Motivated by this result, we adopt a heuristic approach for the selection of γ. We interpret the parameter 2γ as the difference in characteristic scalings between r_i and $\tilde{r} = \sum_{i=1}^N r_i$. Here, we approximate the distributions of r_i and $\tilde{r} = \sum_{i=1}^N r_i$ with Gaussians and adopt their $1 - \alpha$ levels as their characteristic scalings. Then we set γ to be a half of their difference.

5 Simulation

In this section, we present numerical results of the proposed method both in a synthetic setting and using real world datasets.

5.1 Synthetic Experiment

The aim of this experiment is to evaluate the common substructure detection performance of the proposed method. For the sake of comparison, we adopted GLasso [11] as discussed in Section 2.2 and multi-task structure learning (MSL) [13] from Section 2.4 as baseline methods. Since neither method was designed for common substructure learning, we thresholded the variation in the estimated precision matrices $\hat{\Lambda}_1, \hat{\Lambda}_2, \ldots, \hat{\Lambda}_N$ and heuristically extracted the substructure $\hat{\Theta}$ by

$$\hat{\Theta}_{jj'} = \begin{cases} \hat{\theta}_{jj'} \ , & \text{if } \max_{i,i'} |\hat{\Lambda}_{i,jj'} - \hat{\Lambda}_{i',jj'}| < \epsilon \\ 0 \ , & \text{otherwise} \end{cases} \tag{24}$$

where ϵ is some given threshold for the maximum variation. Here, to avoid selecting zero edges as common substructures, we set $\theta_{jj'}$ to zero if $\hat{\Lambda}_{i,jj'} = 0$ for all i and one otherwise.

We generated sparse precision matrices in the following manner. First, we divided d variables $x_1, x_2, \ldots, x_d$ into non-overlapping subsets for each of the N conditions and generated small precision matrices[3] for each subset. In this step, we set some variable subsets and the corresponding matrices to be common to all N conditions so that the substructure could be shared by all GGMs. Finally, we combined these small matrices by adding some edges between them and derived N precision matrices $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$. In the experiment, we set the dimensionality of the data $d = 20$ and the number of conditions $N = 5$. We selected the size of the variable subsets to be 4 and therefore, the generated GGMs were composed of 5 cliques. The resulting GGM structure is shown in Figure 1.

For the simulation, we generated 100 samples according to the Gaussian distribution with Λ_i in each condition and scaled each variable to have a unit variance. We then compared the common substructure detection rates of the three methods. We repeated the simulations for 100 random realizations of the datasets and drew average ROC curves by varying the hyper-parameter ρ as shown in Figure 3. In this experiment, we chose parameter γ from the procedure presented in Section 4.4 with $\alpha = 0.05$. In Figure 3(a), we set the threshold $\epsilon = 10$ for GLasso and MSL, which means that almost all edges were actually treated as common substructures. The resulting curves clearly show that the proposed method outperforms the two baseline methods. If we set ϵ to a smaller value, e.g. $\epsilon = 1$ in Figure 3(b), the ROC curves for GLasso and MSL are no longer monotone increasing for ρ. Here, we note that $\epsilon = 1$ is already a very

[3] We set the diagonal elements in the matrix to one and the off-diagonals elements to a uniformly random value in $[-0.8, -0.1] \cup [0.1, 0.8]$, although this uniformity might be slightly skewed due to the positive definiteness constraint.

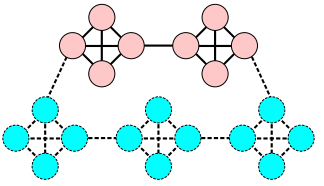


Fig. 1. A GGM structure: edges in the top two cliques (solid lines) are common dependencies, while others are not (dashed lines)

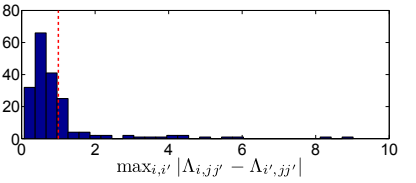
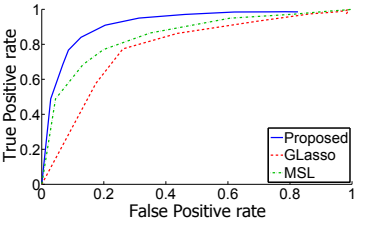
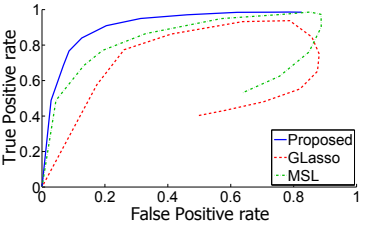


Fig. 2. Histogram of the variation in precision matrices estimated by GLasso with $\rho = 0.0032$. The vertical line denotes the threshold $\epsilon = 1$.



(a) $\epsilon = 10$



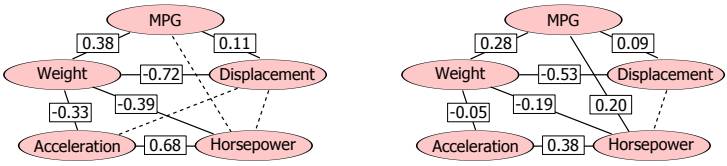
(b) $\epsilon = 1$

Fig. 3. ROC curves. The horizontal axis is the false positive detection rate of common substructures, while the vertical axis is the true positive rate.

optimistic choice. An example of the histogram showing the variation in precision matrices estimated by GLasso with $\rho = 0.0032$ is depicted in Figure 2. In this example, 74% of the estimated non-zero elements have variation less than $\epsilon = 1$ and are judged to be common dependencies. However, only 38% of the true common edges are actually included in the histogram below $\epsilon = 1$, while the other 62% are in the remaining 26% of the estimated non-zero elements. This means that the estimated edge weights using GLasso or MSL for true common substructures vary greatly across the matrices. This example clearly shows the limitation of the existing approaches in that common substructures can easily be masked by estimation variances.

5.2 Analysis of City-Cycle Fuel Consumption Data

We applied the proposed method to the *Auto MPG* dataset from the UCI Machine Learning Repository [20]. The dataset consists of 398 different car data entries containing MPG (Miles Per Gallon), number of cylinders, displacement, horsepower, weight, and acceleration data. Although the name of the car, its model year and the originating region are included in the data, we discarded these fields since they seem to be irrelevant to the other variables. We rearranged the data according to the number of cylinders, giving 199 entries for 4 cylinder cars, 83 for 6 cylinders, and 103 for 8 cylinders. We discarded the data for 3 and 5 cylinders since there were only few entries.



(a) $\rho = 0.1$, $\gamma = 0.062$ (b) $\rho = 0.2$, $\gamma = 0.062$

Fig. 4. Estimated dependency structures for MPG data. The solid lines denote common relations among cars with different numbers of cylinders while the dashed lines are varying dependencies. The numbers attached to solid lines denote the common edge weights.

We applied the proposed method to the 3 datasets containing data for cars with different numbers of cylinders. Each dataset was composed of 5 variables. Empirically, the number of cylinders is closely related to the displacement and the horsepower. The aim of the analysis was to find relations between variables that are irrelevant to the number of cylinders, which might be related to the underlying functional mechanism of cars. As pre-processing, we scaled each variable to have a unit variance.

Figure 4 shows the results for the two settings, $\rho = 0.1$ and 0.2. We chose γ based on the proposed heuristic. In the estimated graph, there are two major cliques composed of *weight, horsepower and acceleration* and *MPG, weight and displacement*, respectively. In the first clique, the relations between mass (weight), acceleration, and force (horsepower) are those expressed by Newton's motion equation. Since each variable has been scaled to unit variance, it is natural that the relation between them is steady. Data fields in the second clique, we believe, they are related to the quality of the car. Typically, expensive cars have many more features including a high specification engine which results in greater weight, higher displacement, and improved MPG. What the results suggest is that this tendency is common to cars with any number of cylinders. We conclude that the proposed method successfully found some reasonable common relations between variables without using any prior knowledge about the datasets.

5.3 Application to Anomaly Detection

In this section, the proposed method is applied to an anomaly detection problem. The task is to identify contributions of each variable to the difference between two datasets. Correlation anomalies [5], or errors on dependencies between variables, is known to be difficult to detect using existing approaches especially with noisy data. To overcome this problem, the use of sparse precision matrices was proposed by Idé et al. [5], since the sparse approach reasonably suppresses the pseudo correlation among variables caused by noise and improves the detection rate. Here, we propose to use the common substructure learning approach. There is a clear indication that the proposed method can further suppress the variation in the estimated matrices. In particular, we expect that dependency structures among healthy variables are estimated to be common, which reduces the risk that such variables are mis-detected and only anomalies are enhanced.

	best AUC	ρ
Proposed	**0.97**	0.05
GLasso	0.96	0.20
MSL	**0.97**	0.05

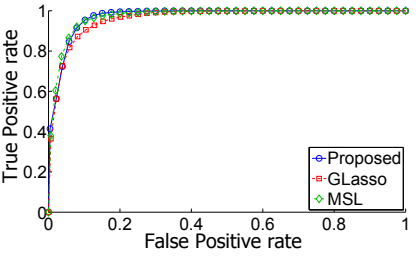
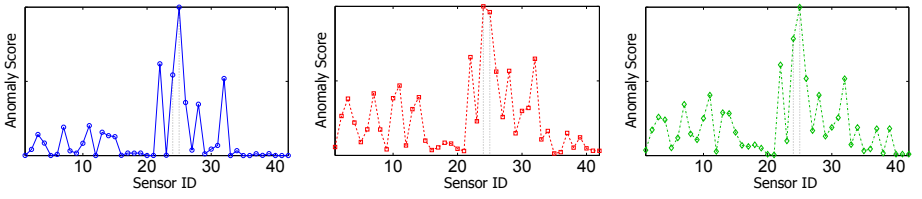


Fig. 5. Anomaly detection : best AUC values and corresponding ROC curves



(a) Proposed Method (b) GLasso (c) MSL

Fig. 6. Anomaly scores. All plots are normalized so that their maximum values are the same. Dotted lines denote true faulty sensors.

We evaluated the anomaly detection performances using the *sensor error* data [5]. The dataset comprised 42 sensor values collected from a real car in 79 normal states and 20 faulty states. The fault was caused by mis-wiring of the 24-th and 25-th sensors, resulting in correlation anomalies. We compared three methods, GLasso, MSL and our proposed method with the anomaly score proposed by Idé et al. [5] which is based on the KL-divergence between two datasets. Since sample covariances are rank deficient in some datasets, we added 10^{-3} on their diagonal to avoid the singularity. For simulation, we randomly sampled 20 datasets from the normal states and 5 datasets from the faulty states, and estimated sparse precision matrices with each method. We set the weight t_i in MSL and the proposed method as $t_i = \frac{1}{40}$ for normal datasets and $t_i = \frac{1}{10}$ for faulty datasets to balance the effects from the two states. Since the anomaly score was designed only for a pair of datasets, we calculated anomaly scores for each of 20×5 pairs and reported the average score and detection rate. We tested each method by varying the parameter ρ between 0.05 and 0.30.

We repeated the above procedure 100 times and drew ROC curves of the average anomaly detection rate with the best area under curve (AUC) results shown in Figure 5. First, we see that MSL and the proposed method surpass the detection rate of GLasso. This is because these two methods estimate precision matrices with joint regularizations. This reduces the estimation variance among matrices while GLasso conducts the estimation separately resulting in more varied estimators, which masks the correlation anomalies. Secondly, though the detection performances are competitive between MSL and the proposed method,

we can see further differences in the resulting anomaly scores in Figure 6. Clearly, the scores for the proposed method show lower significance for normal variables, especially for variables from 16 to 21 and 33 to 42, whereas anomaly variables are still enhanced. This is what we expected in the beginning; that is, the proposed method successfully reduces the nuisance effects and highlights only those variables with correlation anomalies. The remaining peaks at some normal variables are caused by the effect of the two faulty variables, since the correlation anomaly is calculated as faults of a pair of variables.

6 Conclusion

In this paper, we formulated the common substructure learning problem of multiple GGMs and presented an optimization algorithm based on the block coordinate descent. We further showed that the subproblem of the block coordinate descent has three types of solutions and can be solved efficiently with techniques for Lasso and the continuous quadratic knapsack problem. Numerical results on synthetic and real world datasets indicated the clear advantage of the proposed method over existing GGM structure learning methods.

Several future works have been identified: the analysis of the asymptotic property of (7), and the extension of the current formulation to the adaptive Lasso [21] type one to guarantee the *oracle property* [21] of the estimator. Applying the notion of commonness to more general dependency models is also an important work, e.g. non-linear relations or the commonness based on higher order moment statistics.

Acknowledgment. This work was partially supported by a JSPS Grant-in-Aid for Scientific Research(B) #22300054. The authors would like to thank Tsuyoshi Idé and his colleagues for providing the *sensor error* datasets for our simulation. We also acknowledge the many helpful comments from Shohei Shimizu.

References

1. Baillie, R.T., Bollerslev, T.: Common stochastic trends in a system of exchange rates. The Journal of Finance 44(1), 167–181 (1989)
2. Zhang, B., Li, H., Riggins, R.B., Zhan, M., Xuan, J., Zhang, Z., Hoffman, E.P., Clarke, R., Wang, Y.: Differential dependency network analysis to identify condition-specific topological changes in biological networks. Bioinformatics 25(4), 526–532 (2009)
3. Varoquaux, G., Gramfort, A., Poline, J.B., Thirion, B.: Brain covariance selection: better individual functional connectivity models using population prior. Arxiv preprint arXiv:1008.5071 (2010)
4. Ahmed, A., Xing, E.P.: Recovering time-varying networks of dependencies in social and biological studies. Proceedings of the National Academy of Sciences 106(29), 11878–11883 (2009)
5. Idé, T., Lozano, A.C., Abe, N., Liu, Y.: Proximity-based anomaly detection using sparse structure learning. In: Proceedings of the 2009 SIAM International Conference on Data Mining. SIAM, Philadelphia (2009)

6. Lauritzen, S.: Graphical models. Oxford University Press, USA (1996)
7. Dempster, A.P.: Covariance selection. Biometrics 28(1), 157–175 (1972)
8. Meinshausen, N., Bühlmann, P.: High-dimensional graphs and variable selection with the lasso. The Annals of Statistics 34(3), 1436–1462 (2006)
9. Yuan, M., Lin, Y.: Model selection and estimation in the gaussian graphical model. Biometrika 94, 19–35 (2007)
10. Banerjee, O., El Ghaoui, L., d'Aspremont, A.: Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. The Journal of Machine Learning Research 9, 485–516 (2008)
11. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. Biostatistics 9(3), 432–441 (2008)
12. Zhang, B., Wang, Y.: Learning structural changes of gaussian graphical models in controlled experiments. In: Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (2010)
13. Honorio, J., Samaras, D.: Multi-task learning of gaussian graphical models. In: Proceedings of the 27th Conference on Machine Learning (2010)
14. Chiquet, J., Grandvalet, Y., Charbonnier, C.: Sparsity with sign-coherent groups of variables via the cooperative-lasso. Arxiv preprint arXiv:1103.2697 (2011)
15. Guo, J., Levina, E., Michailidis, G., Zhu, J.: Joint estimation of multiple graphical models. Biometrika 98(1), 1–15 (2011)
16. Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., Knight, K.: Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society: Series B 67(1), 91–108 (2005)
17. Caruana, R.: Multitask learning. Machine Learning 28(1), 41–75 (1997)
18. Bach, F.R.: Consistency of the group lasso and multiple kernel learning. The Journal of Machine Learning Research 9, 1179–1225 (2008)
19. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. Journal of Optimization Theory and Applications 109(3), 475–494 (2001)
20. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
21. Zou, H.: The adaptive lasso and its oracle properties. Journal of the American Statistical Association 101(476), 1418–1429 (2006)

Appendix

Proof of Theorem 1: The non-differentiable term in (7), i.e., the regularization term, is continuous and convex, and is a sum of $\mathcal{O}(d^2)$ terms where each term is composed of variables $\Lambda_{1,jj'}, \Lambda_{2,jj'}, \ldots, \Lambda_{N,jj'}$. Moreover, (7) is continuous in a compact level set. Then, the claim follows from Theorem 4.1 in [19]. □

Proof of Theorem 2: We prove this for the case $\|\tilde{\boldsymbol{\xi}}\|_1 > \rho + 2\gamma$, otherwise $\tilde{\xi}$ is a solution to (14) and the claim holds. Let f be the objective function in (14) and $\boldsymbol{\xi}_0$ be one of the feasible solutions. Then, for $\boldsymbol{\xi}'_0 = \boldsymbol{\xi}_0 + \epsilon(\tilde{\boldsymbol{\xi}} - \boldsymbol{\xi}_0)$ with $0 < \epsilon \leq 1$, $f(\boldsymbol{\xi}'_0) \leq f(\boldsymbol{\xi}_0)$ holds from the convexity of f. Therefore, $\boldsymbol{\xi}'_0$ is a better solution to problem (14) as long as $|\mathbf{1}_N^\top \boldsymbol{\xi}'_0| \leq \rho$ and $\|\boldsymbol{\xi}'_0\|_1 \leq \rho + 2\gamma$ are satisfied. The first condition always holds because $|\mathbf{1}_N^\top \boldsymbol{\xi}'_0| \leq (1-\epsilon)|\mathbf{1}_N^\top \boldsymbol{\xi}_0| + \epsilon|\mathbf{1}_N^\top \tilde{\boldsymbol{\xi}}| \leq \rho$. On the other hand, the latter condition $\|\boldsymbol{\xi}'_0\|_1 = \sum_{i=1}^N |\xi_{0,i} + \epsilon(\tilde{\xi}_i - \xi_{0,i})| \leq \rho + 2\gamma$ is no longer valid if $\|\boldsymbol{\xi}_0\|_1 = \rho + 2\gamma$ and $\mathrm{sgn}(\xi_{0,i}) = \mathrm{sgn}(\tilde{\xi}_i - \xi_{0,i})$, which results

in $\tilde{\xi}_i \xi_{0,i} \geq 0$. This is the necessary condition for the solution to (14). Otherwise, we can always improve the solution by the above procedure which contradicts its optimality. □

Proof of Proposition 1: Here, we use the alternative expression of (7):

$$\max_{\Theta, \{\Omega_i\}_{i=1}^N} \sum_{i=1}^N t_i \, \ell(\Theta + \Omega_i; \hat{\Sigma}_i) - \sum_{j \neq j'} \left(\rho |\Theta_{jj'}| + \tilde{\gamma} \max_i |\Omega_{i,jj'}| \right)$$
$$\text{subject to } \Theta + \Omega_1, \ \Theta + \Omega_2, \ \ldots, \ \Theta + \Omega_N \succ 0 \tag{25}$$

where $\Lambda_i = \Theta + \Omega_i$ and $\tilde{\gamma} = \rho + 2\gamma$. The equivalence can be proved by comparing their dual problems. In the bivariate case, let matrices Θ, Ω_i and $\hat{\Sigma}_i$ be

$$\Theta = \begin{bmatrix} 0 & \theta \\ \theta & 0 \end{bmatrix} , \quad \Omega_i = \begin{bmatrix} u_i & \omega_i \\ \omega_i & v_i \end{bmatrix} , \quad \hat{\Sigma}_i = \begin{bmatrix} a_i & r_i \\ r_i & b_i \end{bmatrix} .$$

Since $\sum_{i=1}^N t_i |\omega_i| \leq \max_i |\omega_i|$, the objective function (25) is upper-bounded by

$$\mathcal{L}(\theta, \{u_i, v_i, \omega_i\}_{i=1}^N; \{r_i\}_{i=1}^N) = \sum_{i=1}^N t_i \left\{ \log \left(u_i v_i - (\theta + \omega_i)^2 \right) \right.$$
$$\left. - (a_i u_i + b_i v_i) - 2(r_i \omega_i + \tilde{\gamma} |\omega_i|) \right\} - 2 \sum_{i=1}^N t_i r_i \theta - 2\rho |\theta| \, . \tag{26}$$

Moreover, this coincides with (25) if $\omega_i = 0$ for all i. Therefore, if $\omega_i = 0$ $(1 \leq i \leq N)$ is a maximizer of $\mathcal{L}$, it is also the solution to (25). From the derivative of $\mathcal{L}$ over ω_i, we get that $\omega_i = 0$ is a maximizer if

$$-(\tilde{\gamma} + r_i) \leq \frac{\theta}{u_i v_i - \theta^2} \leq (\tilde{\gamma} - r_i) \, . \tag{27}$$

This is a sufficient condition for (25) to have $\omega_i = 0$ $(1 \leq i \leq N)$ as its optimal value. If $|r_i| \leq \tilde{\gamma}$ holds for all i, the problem (25) coincides with the ℓ_1-regularized maximum likelihood (2) with the above constraints on θ:

$$\max_{\theta} \ \log \left(\tilde{u} \tilde{v} - \theta^2 \right) - \left(\tilde{a} \tilde{u} + \tilde{b} \tilde{v} \right) - 2 \left(\tilde{r} \theta + \rho |\theta| \right)$$
$$\text{subject to } \theta \text{ bounded by (27)} \, , \tag{28}$$

where $\tilde{r} = \sum_{i=1}^N t_i r_i$, and $\tilde{u}, \tilde{v}$ are diagonal components of the resulting common structure. Since the bound of θ involves 0, we see that $\theta = 0$ if $|\tilde{r}| \leq \rho$ from Proposition 1 in [5], and hence $\lambda_i = \theta + \omega_i = 0$. □

Mining Research Topic-Related Influence between Academia and Industry

Dan He

Computer Science Dept., Univ. of California, Los Angeles, CA, 90095-1596, USA
danhe@cs.ucla.edu

Abstract. Recently the problem of mining social influence has attracted lots of attention. Given a social network, researchers are interested in problems such as how influence, ideas, information propagate in the network. Similar problems have been proposed on co-authorship networks where the goal is to differentiate the social influences on research topic level and quantify the strength of the influence. In this work, we are interested in the problem of mining topic-specific influence between academia and industry. More specifically, given a co-authorship network, we want to identify which academia researcher is most influential to a given company on specific research topics. Given pairwise influences between researchers, we propose three models (simple additive model, weighted additive model and clustering-based additive model) to evaluate how influential a researcher is to a company. Finally, we illustrate the effectiveness of these three models on real large data set as well as on simulated data set.

1 Introduction

In recent years, the problem of mining influence in networks, especially social networks, has attracted tremendous attention [5] [7] [9] [10]. In traditional social network, nodes are usually individuals and edges indicate friendship between the pair of individuals. One of the key questions in social network is how ideas, information or influence spreads (cascades) through the network.

Different to traditional social network, in co-authorship network, nodes are researchers and edges indicate the co-author relationship of the pair of researchers. There are weights associated with the edges, indicating the number of publications co-authored by the pair of researchers. Another important difference is there are events, or actions, associated with individuals in the traditional social network. These actions are usually temporal, namely each action has a specific occurrence time. For example, if the action is "purchase ipad", a person and his/her friends in the network may take the action at different times. This temporal property allows quantification of the influences between different individuals and the study of the influence spread model in the traditional social network. The co-authorship network, on the contrary, usually do not have the temporal property. A researcher always publishes papers the same time with his co-authors. Therefore, the influence in the co-authorship network should be defined differently.

In this work, we are specifically interested in the topic-level influence between academia and industry. We believe this is an important problem in that it helps people to evaluate which academia researcher has better connection to a company, and vice

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 17–31, 2011.

versa. This can be very useful in many cases. For example, when a student is seeking an advisor and his career goal is to be a researcher in a company's research lab, he may want to choose an advisor who has tight connection to the company. Another example is funding agencies may choose to award certain type of grants to researchers who work closely with industry companies. Companies may also want to collaborate with academia researchers who have tight industry connection in the same fields. To our knowledge, there is no prior work on mining influence between academia and industry.

A model using Topical Affinity Propagation (TAP) to learn the topic-level social influence on large networks has been proposed recently [13]. Based on the topic-level influence identified by TAP, we proposed three models to mine the topic-level influence of a researcher to a company: (1) simple additive model where we simply sum the influence of the researcher to all the researchers in the company. (2) weighted additive model where we weight the researchers in the company by their internal influence in the company. (3) clustering-based additive model where the researchers in the company are clustered first and then each cluster of closely related researchers (who often publish together) is considered as a "super researcher". We then evaluate the three models on real co-authorship network as well as on simulated co-authorship networks.

2 Related Work

There have been lots of work recently on the problem of mining influence in networks, especially social networks. These work are mainly focused on two main categories of problems: influence probabilities between nodes in the network are pre-defined or these probabilities need to be learned.

For the first category, Domingos and Bichardson [5] studied the viral marketing problem, which targets the most influential users in the network, by viewing the market as a social network and modeled it as a Markov random field. Kempe et al. [7] studied the influence maximization problem, which selects an initial set of users who eventually influence the largest number of users in the social network. A greedy algorithm as well as the first provable approximation guarantees for efficient algorithms are provided. Leskovec et al. [9] modeled the outbreak detection problem as selecting nodes in a network in order to detect the spreading of a virus or information as quickly as possible. Rodriguez et al. [10] developed an approximation algorithm for the problem of identifying optimal diffusion network from temporal data. The algorithm is able to be scaled to large network to trace paths of diffusion and influence through networks and to infer the optimal network that best explains the influence propagation. Chen et al. [4] develop methods to improve the efficiency of the greedy algorithm in [7] to maximize influence. New degree discount heuristics that improve influence spread are further proposed.

For the second category, Goyal et al. [6], studied the problem of learning influence probabilities from historical user action data and try to predict when the users will get activated from the learned probabilities. Saito et al. [11] applied EM algorithm to solve the same problem focusing on the Independent Cascade model of propagation.

Tang et al. [13] introduced a topic-specific social influence problem. Instead of friends, the nodes in the networks are co-authors of one another. Each researcher of the network is associated with a distribution of topics, which are the research topics the

researcher had publication in. Topic models [12] are applied to automatically extract topics from the publications and to initialize the topic distribution of each node. Given a co-authorship network and the topic distribution of the nodes, a topical factor graph [8] is built, in which the observation data are cohesive on both local attributes and relationships. The nodes and edges in the co-authorship network represent the observation data and the relationship in the factor graph, respectively. Three kinds of feature functions are proposed: (1) Node feature function which measures the similarity of the nodes based on their topical similarity or topical interaction strength (using co-authorship information); (2) Edge feature function which measures if there is an edge between the two nodes in the network; (3) Global feature function which measures the representative node on a specific topic. A joint likelihood function is then proposed as the product of the three feature functions for all the nodes in the graph. A Topical Affinity Propagation (TAP) on the factor graph is designed to maximize the likelihood function. The topic specific influence from node a to node b is then defined using a sigmoid function based on two messages in the propagation: how likely node a thinks it influences node b and how likely node b thinks it is influenced by node a on the topic. Therefore, the influence between two nodes are usually not symmetric. Two different propagation frameworks were proposed: a message passing framework and a Map-Reduce framework. We refer to the paper [13] for the details of the model.

3 Models

Since the focus of this work is to model the influence between academia and industry in stead of influence between individual researchers, we first assume we already identify pairwise influences between researchers in the co-authorship network, where the influence needs to be a value of real number. There are multiple methods to identify pairwise influences between researchers. Maybe the most naive method is based on the number of co-authored papers. The influence from author A to author B can be defined as the percentage of their co-authored papers in all the papers published by B. However, this naive method considers a pair of authors as independent to other authors, which is usually not the case. On the contrary, the TAP method [13] applies affinity propagation and therefore is able to better model influences between multiple researchers. In this work, we take the TAP method to estimate the pairwise influences, which are within t the range of [0, 1]. The influences are directed and therefore usually not symmetric. Next we discuss three different models to mine influence from an academia researcher to a company. The influence from industry to academia can be obtained using the same models but with reverse influence direction.

3.1 Simple Additive Model

In this model, we simply sum the topic-specific influence from a researcher to all the researchers in a company. Then we take the sum as the topic-specific influence between the researcher and the company. Therefore, the topic-specific influence from a researcher to a company under *Simple Additive Model* is defined as the following:

$$I_t(r, C) = \sum_{i=1}^{n} I_t(r, C_i) \tag{1}$$

where r is the researcher, C is the company, $I_t(r, C)$ is the influence from r to C for topic t, C_i is the i-th researcher in company C, $I_t(r, C_i)$ is the influence from r to C_i for topic t. Naturally the influence from an university to a company $I_t(U, C)$ is defined as the following:

$$I_t(U, C) = \sum_{i=1}^{n} I_t(U_i, C) \tag{2}$$

where $I_t(U_i, C)$ is the influence from the i-th researcher in university U to the company C. Therefore, the influence from an university to a company is simply the sum of the influence from every researcher in the university to the company.

3.2 Weighted Additive Model

The simple additive model has a problem that all the researchers in the company are weighted equally. This is usually not the case. For example, a manager should have a higher influence in the company than a junior researcher. Therefore, the same influence to the manager and to the junior researcher should not mean equal influence to the company.

To address the above problem, we first compute the internal influence for the researchers in a company. Then each researcher is weighted according to their internal influence. The weights are applied to the researchers when we sum the influences to them. The *Weighted Additive Model* is defined as the following:

$$W_t(C_i) = \frac{\sum_{j=1}^{n} I_t(C_i, C_j)}{\sum_{j=1,k=1}^{j=n,k=n} I_t(C_k, C_j)} \tag{3}$$

$$I_t(r, C) = \sum_{i=1}^{n} I_t(r, C_i) \times W_t(C_i) \tag{4}$$

Where $W_t(C_i)$ is the weight for the i-th researcher in company C for topic t. Similarly, the influence from an university to a company can be computed via Equation 2.

3.3 Clustering-Based Additive Model

The weighted additive model may still have a problem. We show the problem by an example in Figure 1. In this example, A, B are academia researchers. C, D, F, E, G, H are researchers in the same company. Let's assume C, D always publish paper together. Therefore the influences between C, D are relatively high. E influences F, G, H but the influence is lower than the influence between C, D. Therefore, it's possible that the weights of C, D are higher than the weight of E. Thus even though A influences C, D the same extent as B influences E, higher weight of C, D makes A more influential to the company.

However, it may be the case that one of C, D is a senior researcher and the other is a junior researcher. On the contrary, all the researchers E, F, G, H are senior. Therefore, although E has lower influence to F, G, H, E should naturally have a higher influence

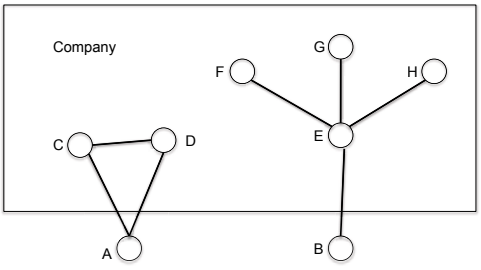


Fig. 1. Example to illustrate the potential issue of weighted additive model

in the company. Thus to determine the weight of a researcher, we may not simply sum the influence of the researcher to others in the same company. Instead, we should consider how the researchers in the company correlates with each other, or how often they publish paper together. By using the correlations, we can cluster the researchers that collaborate a lot into clusters. We then consider each cluster as a *super researcher*. We average the influence in a cluster. Then we can apply the weighted additive model on the clusters instead of on researchers. We call the model *Clustering-based Additive Model*. The benefit of the clustering-based additive model is we address the correlation between researchers in the same company such that the weights on more correlated researchers can be adjusted appropriately.

There are many existing clustering algorithms such as K-means, EM etc.. The problem of these existing clustering algorithms is that the number of clusters is not known and different numbers affect the results of the clustering algorithm. Methods such as modularity based clustering [3] are able to figure out the most likely cluster numbers automatically. However, modularity based clustering only considers the number of nodes in the cluster. It does not consider the distances between the nodes. In our co-authorship network, besides the information that whether the authors collaborate with each other, we also care about how much they collaborate. Therefore the distances between the nodes are important information and should not be ignored. We next present a new method that takes into account the distances between the nodes.

In order to avoid choosing a cluster number, we can build a graph $G = (V, E)$ where each researcher is a node and $e_{i,j} \in E$ iff the similarity between node i and j, $s_{i,j} \geq t$ and t is a similarity threshold. $s_{i,j}$ is defined as the Jaccard's coefficient of the publications of researcher i and j, namely:

$$s_{i,j} = \frac{|P(i) \cap P(j)|}{|P(i)| + |P(j)| - |P(i) \cap P(j)|} \tag{5}$$

where $P(i)$ is the set of publications of researcher i, $|P(i)|$ is the number of publications for set $P(i)$, $P(i) \cap P(j)$ is the set of publications co-authored by researchers i and j. We use co-authorship instead of influence to define the similarity since the co-authorship is symmetric while the influence is directed and not symmetric.

Once we have the graph, we can cluster the nodes in the graph such that the nodes in the same cluster are fully-connected, namely each cluster is a clique. We can apply clique searching algorithms to easily find cliques. However, one problem is a node may

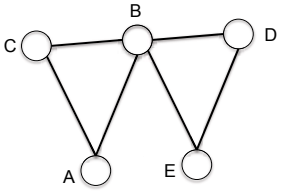


Fig. 2. Example to illustrate a node B is shared by two cliques $\{A, B, C\}$ and $\{B, E, D\}$

Input: A Graph $G = (V, E)$ and a threshold t
Output: A set of clusters $cluster^t$
1. While (there are nodes to be selected)
2. Randomly select a node a
3. Remove a and all the edges associated with a from G
4. assignCluster(a)
5. End
6. Output all the clusters

Fig. 3. Algorithm to generate clusters from a graph given a similarity threshold t

be shared by two possible cliques, as shown in Figure 2. We take a greedy strategy that during the clique searching process, once we assign a node to a clique, we remove the node as well as all the edges connected to the node from the graph. Thus given a threshold t, we can generate a set of clusters $cluster^t$. The algorithm to generate a set of clusters given a threshold t is shown in Figure 3. In line 4, the function assignCluster(node) assigns the node to an existing cluster such that the resulting cluster is still a clique. If there is no such existing cluster, we create a new cluster and assign the node to the new cluster. Since assigning the node to a cluster requires checking the distance of the node to all the nodes in the cluster, the time complexity of assignCluster(node) is $O(n)$. Therefore the complexity of the algorithm is $O(n^2)$ because we need to assign cluster for all the n nodes.

Next we want to determine a good threshold t automatically. We first define an objective distance function as the following:

$$F(t) = || \sum_{k=1}^{k=|cluster^t|} \sum_{i,j \in cluster_k^t} (1 - s_{i,j}) - |cluster^t| || \tag{6}$$

where t is a similarity threshold, $cluster^t$ is the set of clusters when using threshold t, $cluster_k^t$ is the k-th cluster, $s_{i,j}$ is the similarity between nodes i, j, $|cluster^t|$ is the number of clusters. We use $1 - s_{i,j}$ instead of $s_{i,j}$ directly since the more similar two nodes are, the shorter the distance between the two nodes.

The objective function is based on the motivation that we want to minimize the distances of the nodes within the same cluster. Therefore nodes far from each other won't be put in the same cluster. But since obviously the more clusters, or the smaller number of nodes each cluster has, the smaller the distances of the nodes within the same cluster.

If we do not consider the number of clusters, then t=1 will always minimize $F(t)$ since with t=1, the graph contains no edges (we assume the similarity of any pair of nodes is less than 1 and all the nodes will by themselves be a cluster. Thus $F(t)$=0.Therefore, we need to take into consideration the number of clusters. This is similar to how people select a model using AIC or BIC scores [2]. To select a best model, besides minimizing the objective function, the number of the parameters in the model is also taken into account such that the model with smaller parameters is preferred. We noticed that similar ideas have been used in the work of [14].

To find a threshold t to minimize $F(t)$, an extensive search is not applicable since t is continuous. However, in our problem, the number of researchers, or nodes, in the graph is finite. We can compute the pairwise similarity of the nodes in the graph as $s_{1,2}, s_{1,3}, \ldots, s_{n,1}, s_{n,2}, \ldots, s_{n,n-1}$ where n is the total number of nodes. Thus we can try t as $s_{i,j}$ for all i's, j's such that $1 \leq i, j \leq n$ and $i \neq j$ and we show at least one of the $t = s_{i,j}$'s minimizes $F(t)$.

Lemma 1. *At least one of the $s_{i,j}$'s for $1 \leq i, j \leq n$ is able to minimize $F(t)$.*

Proof: We can sort the n^2 $s_{i,j}$'s in ascending order to $s_1, \ldots, s_{n^2}$ then we have $s_k \leq s_{k+1}$ for all $1 \leq k \leq n^2$. For all $s_k < t < s_{k+1}$, the edges in the graph will be exactly the same as the edges in the graph where t=s_k. Therefore, the set of clusters when $s_k < t < s_{k+1}$ will also be identical to the set of clusters when t=s_k and $F(t)$ for both cases are also identical. Thus for any given t, there will be a corresponding s_k such that $s_k < t < s_{k+1}$ and $F(t)$=$F(s_k)$. Thus we can conclude that at least one of the $s_{i,j}$'s for $1 \leq i, j \leq n$ is able to minimize the function $F(t)$. ■

Therefore, the number of trials for t is $O(n^2)$. The overall complexity of our method is thus $O(n^4)$ since for each trial of t, we need to run the algorithm shown in Figure 3 to obtain clusters, whose complexity is $O(n^2)$.

The complexity of our method seems very high, but in reality, n for a company is usually small. The number of trials can be smaller than n^2 as well since the similarity of different pair of nodes can be the same and we only need to try unique values of t. Also most researchers only have a small number of collaborators. If we assume a researcher has constant number $h << n$ of collaborators, instead of $O(n^2)$ edges, the graph has $O(n \times h)$ edges. Therefore the total complexity of the algorithm is $O(n^2 \times h^2)$. Since h is constant, the complexity approximates to $O(n^2)$. So enumerating all $s_{i,j}$'s is not a problem. In our experiments on real data set, the running time is actually in seconds. Once we find a threshold to minimize $F(t)$, we obtain a set of clusters using the algorithm shown in Figure 3. Then to compute the influence of a researcher to a company, we deploy a hierarchical framework.

We first compute the weight of each researcher in every cluster as the following

$$W_t(L_i) = \frac{\sum_{j=1}^{n} I_t(L_i, L_j)}{\sum_{j=1,k=1}^{j=n,k=n} I_t(L_k, L_j)} \tag{7}$$

where L_i is the i-th researcher in the cluster L on topic t, $I_t(L_i, L_j)$ is the influence between L_i and L_j on topic t. We then compute the influence between clusters L and K, $I_t(L, K)$ in the same company on topic t as the following:

$$I_t(L,K) = \sum_{i=1,j=1}^{i=|L|,j=|K|} I_t(L_i,K_j) \times W_t(L_i) \times W_t(K_j)$$

where $W_t(L_i)$ and $W_t(K_j)$ are computed via the Equation 7, $I_t(L_i,K_j)$ is the influence between researchers L_i and K_j on topic t. As we can see, the influence between two clusters are weighted on the researchers in both clusters. We call the influence $I_t(L,K)$ as the *cluster-based* influence. The general procedure to cluster the researchers in a company and to comput the corresponding weights of the cluster in shown in Figure 4. Finally we consider each cluster as a super researcher where the weighted additive model can be applied directly. The influence of each academia researcher r to each cluster L in the company C is computed using the weighted additive model described previously. For each cluster we only consider the researchers in the same cluster and the influence $I_t(r,L)$ is computed in the following way:

$$I_t(r,L) = \sum_{i=1}^{n} I_t(r,L_i) \times W_t(L_i)$$

Thus we compute the influence from a research r in academia to a company C on topic t, $I_t(r,C)$ as the following:

$$W_t(C_L) = \frac{\sum_{K \in cluster(C)} I_t(C_L,C_K)}{\sum_{L \in cluster(C), K \in cluster(C)} I_t(C_L,C_K)}$$

$$I_t(r,C) = \sum_{L \in cluster(C)}^{n} I_t(r,C_L) \times W_t(C_L)$$

where $W_t(C_L)$ is the weight of the cluster L in company C computed via the cluster-based influence.

Input: A Graph $G = (V,E)$ for researcher in a company
Output: A set of clusters and their weights
1. Compute the optimal similarity threshold
2. Generate clusters given the threshold
3. For each cluster
4. compute the weights of the researchers in the cluster
5. End
6. Compute the weights of each cluster
7. Return the clusters and their corresponding weights

Fig. 4. Procedure to cluster the researchers in a company and to compute the weights of the clusters

4 Experimental Results

4.1 Experiments Settings

We use the same data set used by Tang et al. [13]. There are totally eight different topics. The number of researchers in each topic is shown in Table 1. Notice the same researcher may have multiple research areas. We also show the number of companies and universities working on the topics. Similarly, the same company or university may have multiple research areas. The numbers of companies and universities may not be accurate though since there are researchers with missing affiliation information. We obtain the pairwise topic-specific influence between researchers using the model in [13]. The influences are directed and therefore usually not symmetric. The affiliation information of the researchers is actually annotated in the data set of [13]. And each researcher has only one affiliation in the data set. We just use the researchers with affiliation annotation and ignore the others without such information. As we discussed before, most of the researchers should have much fewer collaborators than the total number of researchers. In the data set, on each topic, we observe the researchers influence on average less than 5 researchers from the same company. Therefore our clustering algorithm runs very fast and is able to finish in seconds.

Table 1. Topics in our data set and the number of corresponding researchers, companies, universities for each topic

Topics	#Researchers	#Companies	#Universities
Data Mining	679	33	99
Machine Learning	976	48	97
Database System	1127	66	116
Information Retrieval	657	49	87
Web Services	400	27	48
Semantic Web	671	38	35
Bayesian Network	554	24	45
Web Mining	348	25	47

4.2 Influence of Academia Researchers to Company

We first show the top-5 most influential researchers to the company 'Microsoft' and 'IBM' on 'Data Mining' under different models. The researchers are ranked by their influences under different models. As we can see, in Table 2, For IBM, the most influential researchers and their corresponding ranks are identical under the three models. This is due to the number of researchers in IBM on Data Mining in the collected data set is relatively small (5 in total). The models did change the influences of these researchers but the ranks of them still remain the same. On the contrary, the rank of the most influential researchers changed for Microsoft under these models. The reason the ranks of some researchers become lower from simple additive model to weighted additive model is because in our data set the researchers at Microsoft that were influenced by these academia researchers have relatively low influence in the company.

When the researchers in the company is clustered, the influence of the researchers who have high influence on the clustered researchers will be changed dramatically.

Table 2. The top-5 most influential researchers to the company Microsoft and IBM on data mining under different models. The researchers are ranked by their influences under different models.

Data Mining (Microsoft)			Data Mining (IBM)		
simple additive	*weighted additive*	*clustering-based additive*	*simple additive*	*weighted additive*	*clustering-based additive*
Jiawei Han	Huan Liu	Clark Glymour	Jiawei Han	Jiawei Han	Jiawei Han
Huan Liu	Clark Glymour	Huan Liu	Philip S. Yu	Philip S. Yu	Philip S. Yu
Xifeng Yan	Michail Vlachos	Michail Vlachos	Michail Vlachos	Michail Vlachos	Michail Vlachos
Clark Glymour	Xuanhui Wang	Padhraic Smyth	Tao Tao	Tao Tao	Tao Tao
Philip S. Yu	Padhraic Smyth	Bing Liu	Ricardo Vilalta	Ricardo Vilalta	Ricardo Vilalta

Table 3. The top-5 most influential researchers to the company Microsoft and IBM on database systems under different models. The researchers are ranked by their influences under different models.

Database Systems (Microsoft)		
simple additive	*weighted additive*	*clustering-based additive*
Calton Pu	Venkatesh Ganti	Sharad Mehrotra
Jiawei Han	Luis Gravano	Jiawei Han
Sharad Mehrotra	Sharad Mehrotra	Jeffrey F. Naughton
Venkatesh Ganti	Jeffrey F. Naughton	Venkatesh Ganti
Jeffrey F. Naughton	Jiawei Han	Luis Gravano
Database Systems (IBM)		
simple additive	*weighted additive*	*clustering-based additive*
Kevin Chen-Chuan Chang	Joseph M. Hellerstein	Joseph M. Hellerstein
Joseph M. Hellerstein	Kevin S. Beyer	Kevin S. Beyer
Renee J. Miller	Renee J. Miller	Min Wang
Kevin S. Beyer	Michael J. Franklin	Kevin Chen-Chuan Chang
Michael J. Franklin	Kevin Chen-Chuan Chang	Michael J. Franklin

For the researchers at Microsoft on data mining, we identified two researchers who publish together frequently. More specifically, one researcher published 39 papers, the other published 118 papers and they co-authored 32 papers. It might be the case that one researcher is relatively senior and the other researcher is relatively junior. We then further identified the title of the researcher that is suspected to be senior and he is indeed a manager of Microsoft. The two researchers are grouped in one cluster and therefore the weights of the two researchers are adjusted appropriately. This leads to the change of the influence for academia researchers who influence these two researchers.

One thing to notice is that our data set actually misses affiliation information for many researchers. Therefore, the rank of the most influential researcher on the companies may not be accurate. But the data set is big enough to show the effectiveness of our models and algorithms.

We also show the top-5 most influential researchers to the company 'Microsoft' and 'IBM' on 'Database Systems' and 'Machine Learning' under different models in Table 3 and 4, respectively. Similarly, for database systems researchers at Microsoft, we identify 4 clusters consist of one senior researcher and one junior researcher. We identified the title of those senior researchers and all of them are manager or principal/senior researcher. We show the number of publications by each of them and the number of their co-authored publications in Table 5. As we can see, the junior researchers published at least half or even 80% of their papers with the senior researchers. Therefore,

Table 4. The top-5 most influential researchers to the company Microsoft and IBM on machine learning under different models. The researchers are ranked by their influences under different models.

Machine Learning (Microsoft)		
simple additive	*weighted additive*	*clustering-based additive*
Brendan J. Frey	Aaron Hertzmann	Aaron Hertzmann
Aaron Hertzmann	Michael I. Jordan	Michael I. Jordan
Andrew McCallum	Andrew McCallum	Andrew McCallum
Michael I. Jordan	Brendan J. Frey	William T. Freeman
William T. Freeman	William T. Freeman	Yoav Freund
Machine Learning(IBM)		
simple additive	*weighted additive*	*clustering-based additive*
Inderjit S. Dhillon	Inderjit S. Dhillon	Inderjit S. Dhillon
Adam R. Klivans	Manfred K. Warmuth	Manfred K. Warmuth
Nader H. Bshouty	Roni Khardon	Roni Khardon
Joydeep Ghosh	Geoffrey J. Gordon	Geoffrey J. Gordon
Manfred K. Warmuth	Gerald Tesauro	Gerald Tesauro

Table 5. The number of publications by the relatively senior researcher and the relatively junior researcher and the number of their co-authored publications for the topic 'database systems' at Microsoft.

Senior	Junior	Co-authored
47	23	11
64	16	9
62	18	11
94	28	22

the clustering-based additive model is able to adjust the weights of the two types researchers. Similar clusters are also observed for the topic 'Machine Learning'.

Due to lack of ground-truth, we do not compare the performance of our clustering algorithm with other clustering algorithm such as modularity based clustering, K-means etc. However, our experimental results clearly show that our method is able to capture the senior-junior groups accurately.

4.3 Influence of Universities to Company

We next show our experiments on the influence of universities to companies. We show the top-5 most influential universities to the company 'Microsoft' on Data Mining and to the company 'IBM' on Database Systems in Table 6. It is fairly hard to determine if a rank is good or not since different people have different criteria and therefore there is even no ground-truth for comparison. What's more, our data set is not complete and lots of affiliation information for researchers is missing. Therefore, we do not report a detailed analysis of our ranking in this work. By simply looking at the ranks, we can see the universities that are well-known for their research in data mining and database systems such as 'wisc', 'uiuc', 'cmu', 'berkeley' etc. are ranked high in their influence to the two companies. Some other universities such as 'uic' also have high ranks. The National Center for Data Mining of University of Illinois Chicago is the member of the Data Mining Group [1] and therefore they have good connections with industry

Table 6. The top-5 most influential schools/research institutions to the company Microsoft on data mining and to the company IBM on database systems under different models. The schools are ranked by their influences under different models.

Data Mining (Microsoft)			Database Systems(IBM)		
simple additive	*weighted additive*	*clustering-based additive*	*simple additive*	*weighted additive*	*clustering-based additive*
uiuc	asu	ucr	berkeley	wisc	wisc
cmu	ucr	cmu	wisc	berkeley	berkeley
uic	cmu	asu	uiuc	toronto	toronto
asu	uic	uic	toronto	umd	umd
ucsb	uiuc	uci	umd	uiuc	uiuc

companies. Again, the rank is completely based on our current dataset and may not be accurate due to the missed affiliation information.

4.4 Simulated Data

Due to lack of ground-truth and missing affiliation information, we are not able to validate and compare the three models we proposed on the real data set. Thus we conduct the following set of experiments on simulated data. For simplicity, we only consider one topic. In our simulation, we generate a company with 200 researchers, which is comparable to the number of researchers of the real companies. We randomly select 20 of them as managers who are relatively influential in the company. For each manager, we assign the remaining researchers to his/her group randomly and we set up a *significant influence threshold* and a *low influence threshold* as 0.5 and 0.2, respectively. We assume the managers have significant influence to their group members thus we randomly generate influence from the managers to the other group members. The influences are in the range of [0.5, 1]. Each manager and his/her group then naturally represent a cluster.

Then we generate 400 researchers in the academia. Our ground-truth is we have two types of researchers in academia that have high influence to the company: (1) the researchers in academia who influence many researchers of the company (we call this type of researchers *influence many* researchers). (2) the researchers in academia who influence "important" researchers, namely managers of the company (we call this type of researchers *influence important* researchers). For type one researchers, we set up an *influence many threshold* as 30, where these researchers influence at least 30 researchers of the company. The influences, however, are all below the low influence threshold 0.2. For type two researchers, we set up an *influence important threshold* as 3, where these researchers influence at least 3 managers. The influences are above the significant influence threshold 0.5 and below 1.

We also generate another group of 20 academia researchers who influence researchers in the same group of the company. We call this set of researcher *influence same group* researchers. The motivation is they neither influence many researchers of the company nor any manager of the company. Therefore they do not belong to the influential researchers to the company. However, they do have significant influence to certain amount of researchers in the same group of the company. This is exactly the same situation as shown in Figure 1. As we discussed before, the simple additive model and the weighted additive model may not be able to distinguish them from the real influential researchers

since both models do not consider the relationship of the researchers of the company. On the contrary, the clustering-based additive model may be able to tell the influence is only on a small group of researchers who collaborate quite often, rather than company-wise influence. To validate this, we test the case where the group of researchers collaborating quite often, namely they have significant influence to each other as well as the case the group of researchers collaborating less often, namely they have low influence to each other.

The reason that we choose the above parameter settings is that these parameters are able to illustrate the effectiveness of the clustering-based additive model. With these parameters, both influence many and influence important researchers are indeed very influential, while the influence same group researchers may or may not be influential, depending on the correlation of the researchers in the group. With too extreme parameter settings, such as significant influence threshold as 0.9, or influence important threshold as 10, there might be no chance for the influence same group researchers to be as influential as the two types of researchers who are truly influential.

To compare the performance of difference models, we rank the researchers according to their influence to the company by the three different models we proposed. We then evaluate whether the ranks of the influential researchers to the company are indeed high or not. For the 10 "influence many" researchers and the 10 "influence important" researchers, we expect them to be ranked as the top-20 most influential researchers. Therefore if their ranks drop below 20, we think the model makes errors. As to the 20 "influence same group" researchers, we expect them to be ranked below the "influence important" and the "influence many" researchers. Therefore if their ranks are above 20, we think the model makes errors. We randomly simulate 10 data sets and show the averaged number of errors by each model for each type of researchers. The results are shown in Figure 5.

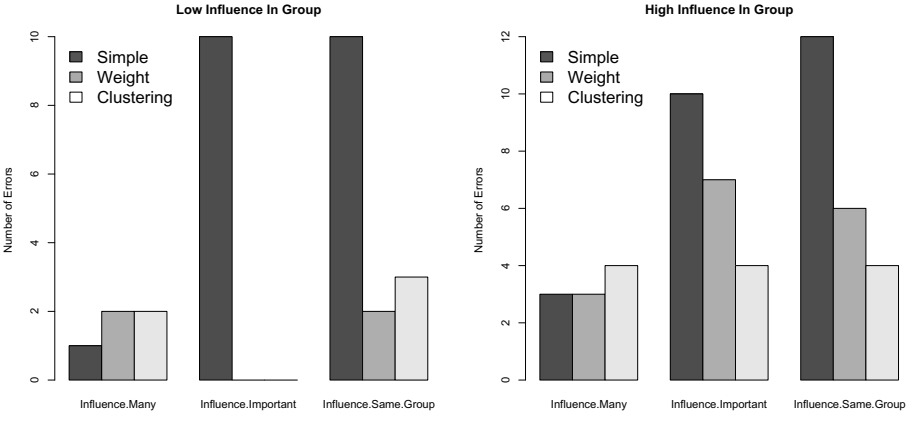


Fig. 5. Comparison of the performance of the three models

As we can see, when the group of researchers have low influence to each other, the ranks of the "influence many" researchers are all good for all three models. The ranks of the "influence important" researchers are good for the weighted additive model and the clustering-based additive model. The ranks are bad for the simple additive model since the "influence important" researchers only influence three researchers in the company, and the model doesn't consider the importance of these researchers. The ranks of the "influence same group" researchers are bad for the simple additive model as expected since they do not influence many researchers. The ranks for the weighted additive model and the clustering based additive model are comparable and are both good.

When the group of researchers have significant influence to each other, the performance of all three models drop for all three types of researchers since it becomes harder to distinguish the "influence same group" researchers with the "influence many" and the "influence important" researchers. For the "influence many" researchers, the ranks of all three models are still ok. For the "influence important" researchers, the ranks of the simple and weighted additive models are both bad since both models think the "influence same group" researchers are really influential to the company. The clustering-based additive model, on the contrary, integrates the information that the researchers of the company being influenced are from the same group and collaborate with each other quite often, and thus obtain relatively better ranks. For the same reason, the clustering-based additive model again achieves the best performance for the "influence same group" researchers.

As a conclusion, the simple additive model tends to assign high ranks to the "influence many" researchers, or the researchers who influence many researchers of the company and the influences are not necessarily significant. The weighted additive model tends to assign high ranks to the "influence important" researchers, or the researchers who influence only a few but important researchers of the company. The clustering-based additive model is not very different from the weighted additive model if the researchers of the company within the same group do not have significant influence to each other. However, when these researchers do have significant influence to each other, the clustering-based additive model has higher accuracy to assign relatively high ranks to the "influence important" researchers and relatively low ranks to the "influence same group" researchers.

5 Conclusion

In this work, we addressed the problem of mining research topic-specific influence between academia and industry. Based on the influence between individual researchers, we proposed three models – simple additive model, weighted additive model, clustering-based additive model – to learn the influence of individual researcher in academia to a company on specific research topics. We further derived the topic-specific influence from a research institution to a company. The influence from industry to academia can be obtained using the same models but with reverse influence direction. In our future work, we'd like to manually complete the missing affiliation information of researchers such that our experimental results are more accurate and the ranks we obtained are more meaningful.

Acknowledgements. The author wants to thank Dr. Tang for sharing the data set used in the work of [13].

References

1. Data Mining Group (2011), http://www.dmg.org/
2. Akaike, H.: A new look at the statistical model identification. IEEE Transactions on Automatic Control 19(6), 716–723 (2002)
3. Brandes, U., Delling, D., Gaertler, M., et al.: On modularity clustering. IEEE Transactions on Knowledge and Data Engineering, 172–188 (2007)
4. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 199–208. ACM, New York (2009)
5. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 57–66. ACM, New York (2001)
6. Goyal, A., Bonchi, F., Lakshmanan, L.V.S.: Learning influence probabilities in social networks. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 241–250. ACM, New York (2010)
7. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146. ACM, New York (2003)
8. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 47(2), 498–519 (2001)
9. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., Glance, N.: Cost-effective outbreak detection in networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, p. 429. ACM, New York (2007)
10. Rodriguez, M.G., Leskovec, J., Krause, A.: Inferring networks of diffusion and influence. In: KDD 2010 (2010)
11. Saito, K., Nakano, R., Kimura, M.: Prediction of information diffusion probabilities for independent cascade model. In: Knowledge-Based Intelligent Information and Engineering Systems, pp. 67–75. Springer, Heidelberg (2010)
12. Steyvers, M., Griffiths, T.: Probabilistic topic models. In: Handbook of latent semantic analysis, p. 427 (2007)
13. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 807–816. ACM, New York (2009)
14. Wu, X., Zhang, C., Zhang, S.: Database classification for multi-database mining. Information Systems 30(1), 71–88 (2005)

Typology of Mixed-Membership Models: Towards a Design Method

Gregor Heinrich

University of Leipzig + vsonix GmbH
Darmstadt, Germany
gregor@arbylon.net

Abstract. Presents an analysis of the structure of mixed-membership models into elementary blocks and their numerical properties. By associating such model structures with structures known or assumed in the data, we propose how models can be constructed in a controlled way, using the numerical properties of data likelihood and Gibbs full conditionals as predictors of model behavior. To illustrate this "bottom-up" design method, example models are constructed that may be used for expertise finding from labeled data.

1 Introduction

In many areas of data mining, it is of interest to re-enact the structure that exists or is assumed in the data by a model that then quantifies this structure for analysis purposes. A good example is knowledge discovery in social community data. Such data often exist in the form of text in documents, which have associated with them meta-data like annotations and ratings, comments and tags, as well as or relational information like authorship, citation and linkage on the Web.

Analysis of such data (that in similar structure arise in other fields, from bioinformatics to computer vision) has often been associated with latent-variable models, and one specific type of such models has empirically led to robust results in the presence of sparsity and noise in the data and especially with complex interrelations between items of different modalities. These latent-variable models cover mixed membership, that is, each document etc. may be a member of a mixture of latent variables, which themselves may be interpreted as a "topic", a group of items/words etc. of similar meaning. Simple models of this model family (also referred to as topic models) were based on handling co-occurrence between words in documents, as in latent Dirichlet allocation (LDA) [1], or words associated to authors, as in the author–topic model (ATM) [2], and these seminal approaches have been extended into various directions.

Typically, in the literature such models are designed by assuming generative processes to re-enact observations, for example each word in LDA is thought to be generated by sampling a topic indicator from a document-specific topic multinomial and a word from a topic-specific vocabulary multinomial.

While this viewpoint of generative processes is intuitive in the sense of explaining models, it remains somewhat "short-sighted" in terms of the connection of model structures and behavior to data structures: The actual behavior of data likelihood (as an

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 32–47, 2011.

essential objective measure of model quality given trained parameters) is not directly found from the model structure. So is the structure of the inference equations necessary to find the optimum model parameters, typically by running approximative EM-type optimization.

On the other hand, meanwhile generic formulations of mixed-membership/topic models have been proposed, such as [3] and [4], that derive numerical properties across particular models and may allow some deeper look into correspondence between model and data structure.

Objectives and Outline. In this article, we complement the pure Bayesian-network viewpoint adopted in the literature by simplifying model structures. We aim at using these structures as building blocks to construct models in a principled way, keeping track of the model behavior when assembling the pieces to fit to data structures in question.

In particular, we will give a deeper introduction of latent-variable models in Sec. 2, reviewing a generic formulation. Based on this, we present a typology of model substructures in Sec. 3 that will serve as the basis for model construction in Sec. 4. Finally, we present a brief empirical study of the proposed approach in Sec. 5.

2 Networks of Mixed Membership

In [3], a generic view on topic models has been taken that formulates their structure as what we may call here "networks of mixed membership" (NoMMs). A NoMM is a directed acyclic graph whose nodes represent sets of mixture components and whose edges transmit variables to child nodes. Selection of components is achieved as a function of the incoming edge values, and values sampled from selected components are transmitted to child nodes. This process ultimately leads to observations at one or more terminal nodes.

Graphically, a simple NoMM structure is shown in Fig. 1, using the seminal LDA model as an example and introducing generic quantities. By default, NoMMs are Bayesian mixture models whose nodes include component parameters along with their prior distributions, and in the typical case, components, with index $k^\ell \in [1, K^\ell]$, have multinomial parameters, $\vec{\vartheta}^\ell_k$, with conjugate Dirichlet priors with hyperparameters $\vec{\alpha}^\ell$. Edges represent variables, $x^\ell_i \in [1, T^\ell]$, that run along sequences, $i^\ell \in I^\ell$. In this notation, a superscript like $\cdot^\ell$ indicates a "mixture level" in the network (a node with its direct child edges) and by convention also extends to variable indices, that is, $x^\ell_i \equiv x^\ell_{i^\ell}$.

Opposed to Bayesian networks (BNs [5]), the NoMM representation strictly distinguishes model variables (that grow with the data) and model parameters (that control variable generation), representing them as edges and nodes, respectively. In connection to this, two types of BN plates are distinguished: On one hand, "sequence plates" run over the data points and correspond to NoMM sequence indices i^ℓ as part of the data "streamed" along edges, x^ℓ_i. On the other, "component plates" index mixture components. In NoMMs, they correspond to the indices k^ℓ in nodes, which depend on incoming information as arguments of component selection functions, $k^\ell = f^\ell_k(\text{parents}(x^\ell_i), i^\ell)$.

The resulting structure is a domain-specific compact alternative to BNs that directly visualises the flow of (typically discrete) information through the generative model,

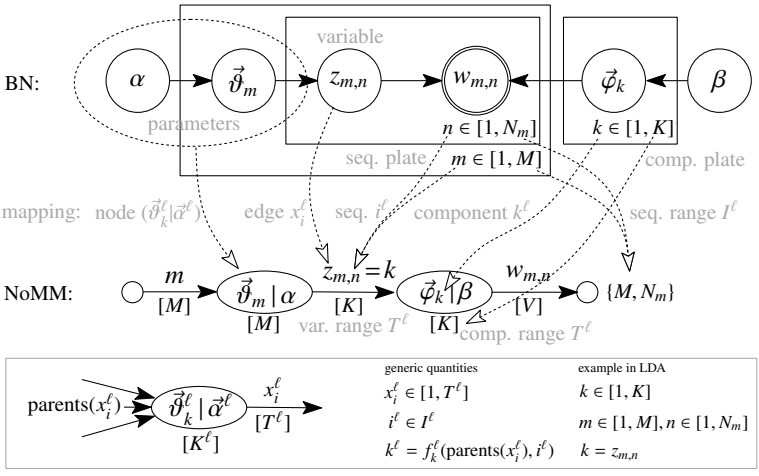


Fig. 1. NoMM notation and correspondence to Bayesian network for example model LDA and generically

mimicking a "systems view" with the nodes representing sub-systems and the edges signals processed by them. Clearly, the NoMM representation is focussed on the domain of mixture models, and especially such models that use complex interactions between different mixtures, such as mixed-membership models and topic models.

2.1 Numerical Properties

With conjugate distributions in NoMM nodes, there exist closed-form solutions for approximate Bayesian inference that lead to good empirical results [1,6], and for collapsed Gibbs sampling [3] and variational inference [4] meta-algorithms have been proposed that may be specialised to a wide range of models. Due to its relatively simple forms [3], especially Gibbs sampling may bear some intuitive meaning.

For the following considerations, let upper-case symbols denote sets of their lower-case counterparts introduced above. That is, Θ, A, and X correspond to all component parameters, hyperparameters and variables of a given model. If a superscript $\cdot^\ell$ is given, symbols are specific to a level. Among variables, X, we further distinguish the sets of hidden and visible variables, H and V.

Posterior. Aside from being the key to model training, the posterior, $p(H, \Theta|V, A)$, may provide insight into the expected behaviour of a model. In a collapsed Gibbs sampler as used for LDA-like models [6,7], the posterior is represented by *full conditional distributions*, the marginals of hidden variables at single data points i, $p(h_i|V, H_{-i}, A)$, given all other information except the parameters Θ, which are integrated out in collapsed inference. Here the index $-i$ denotes exclusion of i. The set of these distributions forms the transition matrix of a Markov chain, and round-robin sampling through i over time leads to a stationary state that simulates the true posterior. Full conditionals may therefore be seen as a low-dimensional representations of the true posterior.

In NoMMs, full conditionals have the following form [3]:[1]

$$p(h_i \mid V, H_{-i}, A) \propto \prod_{\ell} \prod_{k^\ell} \frac{\mathrm{B}(\vec{n}^\ell_k + \vec{\alpha}^\ell)}{\mathrm{B}(\vec{n}^\ell_{k,-i} + \vec{\alpha}^\ell)} \tag{1}$$

where $\mathrm{B}(\vec{x})$ is the multidimensional beta function [8] and $\vec{n}^\ell_k = \{n_{kt}\}^\ell_t$ the "co-occurrence" count vector between component index k^ℓ and node output values t^ℓ. Note that h_i are dependent hidden variables for an observation v_i across different levels ℓ.

To illustrate the principle that underlies (1), it may be noted that its factors reduce to simple quotients of sums if the exclusion of the current sample (with $-i$) from the vectors corresponds to a unit difference between numerator and denonimator:

$$p(h_i \mid V, H_{-i}, A) \propto \frac{n^a_{kt,-i} + \alpha^a_t}{\sum_t n^a_{kt,-i} + \alpha^a_t} \cdot \frac{n^b_{kt,-i} + \alpha^b_t}{\sum_t n^b_{kt,-i} + \alpha^b_t} \cdots , \tag{2}$$

that is, the normalised and smoothed co-occurrence counts reinforce the respective sampling weights in a "rich-get-richer" manner, which is known from Pólya urn sampling schemes associated with the Dirichlet–multinomial compound distribution.

Likelihood. Another descriptive property is the likelihood under the set of trained model parameters. Node parameters, $\Theta^\ell = \{\{\vartheta^\ell_{kt}\}_t\}_k$, themselves are simply the expectations of the Dirichlet priors given the co-occurrences, $\vartheta_{kt} \propto n_{kt} + \alpha_t$, and based on this, the likelihood of observations under the model, Θ, may be expressed as:[1]

$$p(v_i|\Theta) = \sum_{h_i} \prod_{\ell} \vartheta^\ell_{kt} \tag{3}$$

where the summation over h_i refers to all configurations of values of the dependent hidden variables.

Model Structure Influence. In the full conditional and the likelihood, the structure of the NoMM and its component selection functions f^ℓ_k control the association of values k^ℓ and t^ℓ of each level with model variables h_i and v_i, corresponding to paths over levels ℓ that assemble the products in (1) and (3). This and the appearance of the intuitive co-occurrence counts in both likelihood and full conditional may be a key for model design. Consequently, we consider building models from network sub-structures.

3 Model Structure

In the following, we study the decomposition of NoMMs into sub-structures, first taking a look at how models are generically decomposed and then at specific sub-structures.

Notation. For notational simplicity, we define a shorthand for the factors in the generic full conditional (1):

$$q(k,t) \triangleq \frac{\mathrm{B}(\vec{n}^\ell_k + \vec{\alpha}^\ell)}{\mathrm{B}(\vec{n}^\ell_{k,-i} + \vec{\alpha}^\ell)} \overset{\text{case of (2)}}{=} \frac{n_{kt,-i} + \alpha_t}{\sum_t n_{kt,-i} + \alpha_t} , \tag{4}$$

[1] This is a simplifying view for clarity, see Appendix A for details.

emphasizing the interrelation of indices that is expected to play a vital role in designing models. We introduce other conventions: Indexes added up with $\oplus$ refer to sums of the indexed counts, for instance $q(a, b \oplus c)$ contains $n_{ab} + n_{ac}$. Furthermore, if hyperparameters, α, are considered explicitly, the notation is augmented to $q(k, t \,|\, \alpha)$; if this information is clear from context, it is omitted to avoid notational clutter.

3.1 Model Decomposition

As a prerequisite to analyzing models, it is of interest to know how they decompose into sub-structures in terms of their full conditional and likelihood functions.

Full Conditional. Decomposing (1), it may be seen that partial full conditionals of sub-structures, $w(\cdot)$, can be factored with other parts of the model: $p(\cdot) = \prod_c w_c(\cdot) = \prod_c \prod_d q_d(\cdot)$. This enables us to indeed look at the sub-structures separately. For example, considering two sub-structures with hidden variables x and y to be connected with a hidden variable b, constructing a full conditional term $w(x, y, b|a, c)$ from the sub-terms $w(x|a, b)$ and $w(y|b, c)$ just multiplies their "q-terms" $q(a, x)q(x, b)$ and $q(b, y)q(y, c)$.

Likelihood. The data likelihood as an indicator of expected model performance (i.e., the best result it can in principle achieve) may like the full conditional be partitioned into substructures. From (3), it may be inferred that the inner terms of the likelihood of the complete model factor into that of sub-models. If two dependent substructures are joined, the marginal sums $\sum_h$ need to be taken care of, which is done by summing over hidden variables that connect the sub-structures. For example, considering two substructures with hidden variables x and y that are to be connected with a hidden variable b, the likelihood becomes: $p(c|a) = \sum_b p(c|b)p(b|a) = \sum_b (\sum_y \vartheta_{b,y}\vartheta_{y,c} \sum_x \vartheta_{a,x}\vartheta_{x,b})$.

3.2 Typology of Sub-structures

In order to analyze the structure of NoMMs usable in practice, we adopted an inductive approach based on an extensive study of the state of the art in topic models. This study resulted in a set of primitive structures that NoMMs are topologically composed of, in particular characterizing these structures (1) according to probability distributions their nodes use, (2) the way how models branch node information, that is, distribute samples of a given node, and finally (3) the way how models merge information, that is, how incoming data index components of a node. For reference, an overview of the described structures is given in Fig. 2, along with the numerical behavior of the Gibbs full conditional and the likelihood of observations. In these quantities, dependencies on A and Θ have been omitted. Although this set of structures is not considered a complete one, it fully covers all of the example models mentioned in this article, except nonparametric ones.

1. Node Types. Besides the standard Dirichlet–multinomial node with hidden parameters used in models like LDA [1](N1; we introduce alphanumeric structure class labels), there are special types that use alternative parameter distributions. First of all, N1 types may have different variants: While N1a uses a single hyperparameter, N1b introduces a selection function for $\vec{\alpha}_j$ that may be used to add an additional level of grouping among components (see App. A).

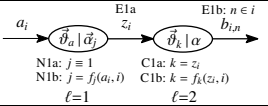
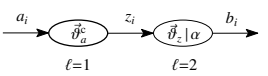
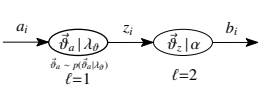
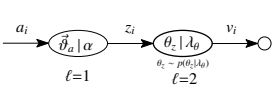
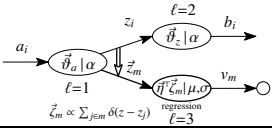
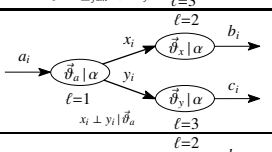
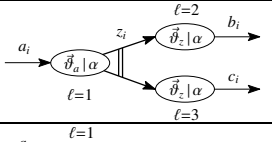
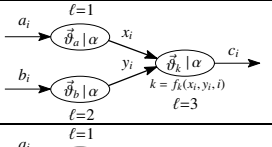
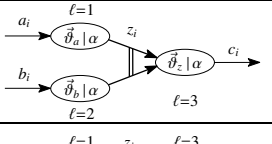
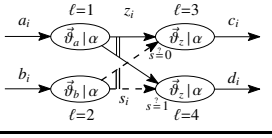
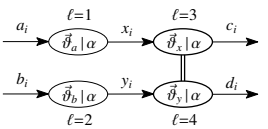
ID. Name	Structure diagram	Gibbs sampler weight w, Likelihood p for single token i Modelled aspect, Example models
N1, E1, C1. Dir–Mult nodes, unbranched		$w(z\|a,b) = q(a,z)q(z,b)$ E1b:$q(a,z)q(z,b_1 \oplus b_2 \oplus \dots b_{N_i})$ $p(b\|a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}$ *Mixture/admixture*: LDA [1], PAM [9]; LDCC [10] (E1b)
N2. Observed parameters		$w(z\|a,b) = \vartheta^{\mathrm{c}}_{a,z}\, q(z,b)$ $p(b\|a) = \sum_z \vartheta^{\mathrm{c}}_{a,z}\vartheta_{z,b}$ *Label distribution*: ATM [2]
N3. Non-Dirichlet prior		$w(z\|a,b;\vec{\vartheta}_a) = p(z_i\|a_i,\vec{\vartheta}_a)q(z,b)$; M-step: estimate $\vec{\vartheta}_a$ [11] $p(b\|a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}$ *Alternative distributions on the simplex*: CTM [11]: $\vec{\vartheta}_a \propto \exp\vec{\eta}$, $\vec{\eta}\sim\mathcal{N}(\vec{\mu},\underline{\Sigma})$; TLM [12]: hierarchy of Dirichlet priors
N4. Non-discrete output		$w(z\|a,v;\theta) = q(a,z)p(v_i\|\theta_z)$; M-step: estimate θ_z $p(v\|a) = \sum_z \vartheta_{a,z}p(v\|\theta_z)$ *Non-multinomial observ.*: Corr-LDA [13], GMM [14]: $p(v\|\theta) = \mathcal{N}(\vec{x}\|\vec{\mu},\underline{\Sigma})$
N5, E4. Regression		$w(z\|\vec{z}_m,v_m,a,b) = q(a,z)\,q(z,w)\mathcal{N}(v_m\,\|\,\vec{\eta}_v^{\mathsf{T}}\vec{\zeta}_m,\sigma^2)$; M-step: estimate $\vec{\eta}_v,\sigma^2\,\|\,\vec{z},\vec{v}$ (for linear regression) prediction: $v_m = \vec{\eta}_v^{\mathsf{T}}\vec{\zeta}_m$ *Regression/supervised learning*: Supervised LDA [15]
E2. Independent edges		$w(x,y\|a,b,c) = q(a,x\oplus y)q(x,b)q(y,c)$ $p(b,c\|a) = \sum_x \vartheta_{a,x}\vartheta_{x,b}\sum_y \vartheta_{a,y}\vartheta_{y,c}$ *Common mixture of causes*: Multimodal LDA [16]
E3. Coupled edges		$w(z\|a,b,c) = q(a,z)q(z,b)q(z,c)$ $p(b,c\|a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}\vartheta_{z,c}$ *Common cause*: Hidden relational model (HRM) [17], Link-LDA [18]
C2. Combined indices		$w(x,y\|a,b,c) = q(a,x)q(b,y)q(k,c)$ $p(c\|a,b) = \sum_x[\vartheta_{a,x}\sum_y \vartheta_{b,y}\vartheta(k,c)]\,, k = f_k(x,y,i)$ *Different correlated causes, relation*: hPAM [19], HRM [17], Multi-LDA [20]
C3. Coupled indices		$w(z\|a,c) = q(a,z)q(z,c\oplus\tilde{c})$, $w(z\|b,c) = q(b,z)q(z,\tilde{c}\oplus c)$ $p(c\|a,b) = \sum_z(\vartheta_{a,z}+\vartheta_{b,z})/2\cdot\vartheta_{z,c}$ *Different causes, same effect*: (proposed here)
C4. Switch		$w(z,s\|a,b,c,d) = q(a,z)[q(b,\mathrm{c})q(z,c)]^{\delta(s,1)}\cdot[q(b,\mathrm{d})q(z,d)]^{\delta(s,2)}$ $p(c,d\|a,b) = \sum_z \vartheta_{a,z}[\vartheta_{b,s=0}\vartheta_{z,c} + \vartheta_{b,s=1}\vartheta_{z,d}]$ *Select complex submodels*: Multi-grain LDA [21], Entity-topic models [22]
C5. Node coupling		$w(x,y\|a,b,c,d) = q(a,x)q(b,y)[q(x,c\oplus d)]^{\delta(x,y)}\cdot[q(y,\tilde{c}\oplus d)]^{1-\delta(x,y)}$ $p(c,d\|a,b) = \sum_x \vartheta_{a,x}\vartheta_{x,c}\sum_y \vartheta_{b,y}\vartheta_{y,d}$ *Correlation of submodels, relations*: Simple relational component model [23], Relational topic model [24]

Fig. 2. NoMM sub-structure types. Notation (also see (4)): $a \oplus b$ adds counts $n_{\cdot a} + n_{\cdot b}$; $\tilde{c}$ means that the count is not decremented with $-i$ in (1).

Other node types vary the distributions they represent: One important type uses observed parameters to accommodate label information, as authorship metadata in the author–topic model [2]. In this case, the prior is lost (N2). Furthermore, there are models with alternative prior distributions (N3), such as "structured" Dirichlet distributions (N3a) [12], and non-conjugate priors for the parameters, such as logistic-normal (N3b) [11]. Varying the output distributions allows modelling of non-discrete observations (N4), for instance using Gaussian with conjugate Gaussian and inverse-Wishart priors, esp. in media mining [13,25]. Another type of non-discrete output may be produced by regression nodes (N5). In connection with aggregation edges (E4, below), N5 nodes apply a regression model to subsets $\vec{z}_m$ of hidden values, $\vec{z}$, allowing supervised learning approaches within the framework of mixed-membership models.

2. Forks / Edge Structures. When connecting nodes of the network, there are different configurations if nodes that receive the output of a given node. The most frequent type of connection is an unbranched edge to a single node (E1). Beside standard unbranched edges (E1a), type E1b incorporates aggregation of a subsequence, such as words as part of sentences: From a single sample of the parent node z_i, e.g., a sentence topic, a whole sub-sequence of tokens $\vec{b}_i$ is produced, as in [10]. To sample z_i, the corresponding Gibbs full conditional term becomes $q(z_i, \vec{b}_i)$, which causes (1) to deviate from its standard form, (2).

Apart from E1 edges, branching edges to several nodes is a common structure. Here either the samples are generated independently (E2), or both children are forced to the same latent variable (E3).

As an interpretation, branching may be seen as a common cause to several observed modalities. Note that the $\oplus$ in the E2 Gibbs weighting function in Fig. 2 expands to:

$$q(a, x \oplus y) = \frac{\mathrm{B}(\vec{n}_a^{(x)} + \vec{n}_a^{(y)} + \alpha)}{\mathrm{B}(\vec{n}_{a,-i}^{(x)} + \vec{n}_{a,-i}^{(y)} + \alpha)} = \frac{(\hat{n}_{ax,-i} + \alpha - \delta(x,y))(\hat{n}_{ay,-i} + \alpha)}{(\sum_t \hat{n}_{at,-i} + \alpha)((\sum_t \hat{n}_{at,-i} + \alpha) + 1)} \tag{5}$$

where $\hat{n}_{at}$ corresponds to the added contributions of both branches and $\delta(x, y)$ the Kronecker delta.

The last edge type (E4) converts a sequence of values to a vector that may be used for instance for regression, thus complementing the regression node type (N5).

3. Joins / Component Selectors. The dual structure type to a fork is a join, a structure in the network that collects edges at the input of a node and computes an index k from the set of incoming values. Such index structures may trivially collect a single edge value as in LDA (C1a), use an edge value and a sequence index as in pachinko allocation models (PAM) [9] (C1b) or be constructed out of several hidden values (C2), as in [19]. Such multi-inputs are made dependent by observed node output and may be used to merge several influences. It is illustrative to verify this by using the information in Fig. 2. The Gibbs weighting term for the C2 structure is $w(x, y|a, b, c) = q(a, x)q(b, y)q(k, c)$, and if the indices x and y simply refer to the dimensions of k, i.e., $k = f_k(x, y) = (x, y)$, one component exists for each combination of input values. With c an observed edge, x and y become dependent, and the full conditional tends to be high wherever $n_{(x,y)c}$ as part of $q(k, c)$ is high. According to the clustering property of the Dirichlet, sampling (x, y) jointly with c further increases $n_{(x,y)c}$.

While branching structures have coupled and independent variants (E3 and E2), so far there seems to exist no structure in literature that does the same for component indices. A desirable structure that complements C2 may take values from multiple inputs and map them into the same variable range. As an approach to this, we propose a substructure that duplicates the sampling process for incoming branches, with the merging node collecting counts from both, and Fig. 2) shows this as structure C3.

Looking at the Gibbs weighting term in Fig. 2 is illustrative. The shorthand $w(z|a,c) = q(a,z)q(z,c \oplus \tilde{c})$ corresponds to:

$$w(z|a,c) = \frac{n_{az,-i} + \alpha}{\sum_z n_{az,-i} + \alpha} \frac{n^{(a)}_{zc,-i} + n^{(b)}_{zc} + \alpha}{\sum_z n^{(a)}_{zc,-i} + n^{(b)}_{zc} + \alpha} \tag{6}$$

and analogously for $w(z|b,c)$.[2] The contributions of both incoming edges, $n^{(a)}_{zc}$ and $n^{(b)}_{zc}$, are summed in the second quotient, effectively superimposing their influences. Compared to C2, this behavior is slightly different: While the effect of a C2 structure is comparable to an intersection of the co-occurrences between pairs (x,c) and (y,c), the C3 structure is likely to behave closer to a union operator.

Another variant of component selector structures is to switch input edges according to the value of a parent node (C4) [21,22]. This allows control of the influence of more complex sub-models in the branches switched.

The final component selector structure results from sharing parameters among different nodes (C5). This allows coupling of different sub-models without additional need for edges and has been of particular interest in analyzing relational data, see, e.g., [23].

4 Towards a Model Design Method

Different to approaches to design models representable by NoMMs, including mixed-membership/topic models, we propose a design method based the "library" of model structures collected in Fig. 2. This method directly takes into account model assumptions and formalizes the actual steps to reach viable structures in a straight-forward workflow. In the following, we outline the general method and subsequently illustrate it with an example design.

4.1 Designing a Design Method

From Sec. 3, we have discussed how the likelihood reflects the general potential of reaching some model quality (likelihood of held-out data is a standard metric in topic modeling), and that Gibbs full conditionals may serve as an indicator of how such an optimization may be achieved: As a low-dimensional "excerpt" of the posterior, a Gibbs sampler will maximize the weights for those latent dimensions that lead to the best model given the data in a Bayesian sense.

Along with any special metrics to capture model quality or computational complexity, these two measures may also be used as predictors of model behavior. For NoMMs, one may develop a design method from them, a strategy to design models is proposed as follows:

[2] For simplicity, we assume they are sampled in two distinct sweeps.

1. Define data *input*: modalities (type of documents, metadata, relational structure) and dimensions available.
2. Define model *output*: results expected, under which *metrics*. This may include retrieval measures and computational complexity.
3. Make *assumptions*: e.g., "topics ⇔ document semantics", "labels ⇔ topics", where "⇔" refers to a correspondence via correlation or co-occurrence. This will be elaborated below.
4. *Structure* model: with artefacts from Fig. 2, map assumptions to structures, often a correspondence,"⇔", leads to a node in the model.
5. *Predict* behaviour: Gibbs + likelihood (Fig. 2) and metrics.
6. *Iterate* model: optionally go to Step 3 or 4.

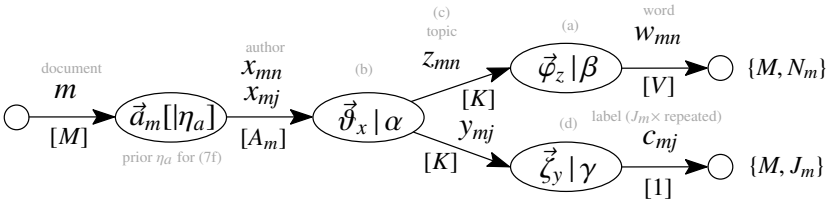
For the design, Step 3 is central, and a rule of thumb is to gather as many assumptions as are needed to "connect" all modalities and hidden assumed structures (like topics). Typically assumptions are qualitative and may be based on correlation or, on the token/item-level, co-occurrence, for instance, terms co-occur in documents and in topics, etc. If topics co-occur in other topics, we obtain a hierarchy of topics. Alternatively, viewpoints of mixing or generative processes may be adopted: Mixing assumes that the output of a node is a mixture of its components, and the generative-process perspective is that used in traditional topic modelling with Bayesian networks. By analogy between BN and NoMM representations (cf. Fig. 1), assumptions can be transformed between them.

Ideally, there exists a one-by-one correspondence between data modalities and assumed hidden structures on one side and model structures on the other. Empirically, the strategy works best with one assumption on each data structure in question, excluding relationships that are transitive, like document ⇔ label = (document ⇔ topic) ∘ (topic ⇔ label). Under-determination of structure by assumptions leaves more structures arbitrary, increasing room for experimentation. For over-determination, assumptions may be prioritized.

For Step 4, some intuition is required to map assumptions to models. Currently, a set of rules is being developed to formalize this process. The next step in this direction is to develop a clearer mapping between the structure types and assumptions, complementing the considerations undertaken on likelihood and full conditional properties in Sec. 3. Furthermore, criteria like scalability are important, as adding any dependent hidden variables increases model complexity considerably: Computational load is on the order of $O(\prod_{h^\ell} T^\ell)$ for dependent h^ℓ.

4.2 Example: Expert–Tag–Topic Model

To illustrate model design, we consider an example scenario: For expert finding, a community of authors is to be indexed to recommend the best expert given a term query or a subject descriptor. While the former may be solved using the author–topic model (ATM) [2], the latter is special to our scenario: Subject descriptors, such as ACM CCS or Medline MeSH, have a controlled vocabulary and are added to the documents authored by experts. For such a scenario, we construct an "expert–tag–topic" (ETT) model using the method above:



$$p(x_{mn}{=}x, z_{mn}{=}z \,|\, w_{mn}{=}w, \{\vec{x}, \vec{z}, \vec{w}\}_{-mn}, \vec{y}) \propto a_{m,x} q(x, z \oplus y) q(z, w) \tag{7a}$$

$$p(x_{mj}{=}x, y_{mj}{=}y \,|\, c_{mj}{=}c, \{\vec{x}, \vec{y}, \vec{c}\}_{-mj}, \vec{z}) \propto a_{m,x} q(x, y \oplus z) q(y, c) \tag{7b}$$

$$p(w_{mn}|\vec{a}_m, c_m, \Theta) = \textstyle\sum_x a_{m,x} \sum_z \vartheta_{x,z} \varphi_{z,w} \qquad w_{mn} \perp c_{mj} \,|\, \Theta \tag{7c}$$

$$p(c_m|a, \Theta) = \textstyle\sum_y \vartheta_{a,y} \zeta_{y,c} \tag{7d}$$

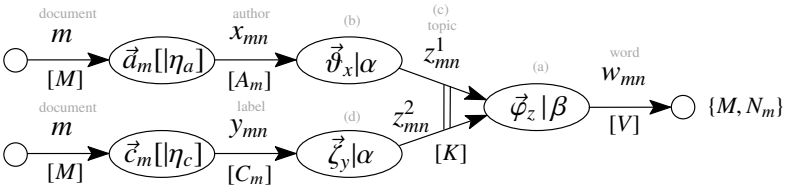
$$p(a|c_q, \Theta) \propto p(c_q|a, \Theta) p(a|\Theta)\,, \quad p(a|\Theta) \propto \textstyle\sum_{mn} \delta(x_{mn} - a) \tag{7e}$$

$$p(a|\vec{w}_q, \Theta) \propto \textstyle\sum_n \delta(x_{qn} - a)\,, \; x_{qn} \sim \text{Gibbs (7a) with } a(\cdot) = q(m, x \,|\, \eta_a) \tag{7f}$$

Fig. 3. Expert–Tag–Topic model, iteration 1

1. *Input*: document text $\vec{w} = \{\{w_{mn}\}_{n=1}^{N_{m=1}}\}_m^M$, subject labels $\vec{c} = \{c_m\}_{m=1}^M$, authorship $\vec{a} = \{\{a_{mx}\}_{x=1}^{A_m}\}_{m=1}^M$.
2. *Output*: expert recommendations $p(a|\vec{w}_q)$ and $p(a|c_q)$ for queries $\vec{w}_q$ and subject labels c_q; *metric*: subjective consistency of topics estimated.
3. *Assumptions*: (a) topics z_{mn} ⇔ document text w_{mn} (semantic similarity of items represented by topics), (b) authors a ⇔ topics $z_{mn} \forall m : a \in \vec{a}_m$, (c) authors a_{mx} ⇔ document text w_{mn}, (d) topics z_{mn} ⇔ labels c_{mj}.
4. *Structure*: Topics z_{mn} appear as the central variable, and we combine on one hand the author–topic model, which fulfills assumptions (a)–(c) and introduces an author–word association x_{mn}, and on the other hand a branch with an N1 node that generates an observable label from a topic, corresponding to assumption (d). The resulting model, "ETT1", is shown in Fig. 3, with the author–topic model in the upper branch and the label branch below. With the E2 branching structure in center, the Gibbs sampling term of the author–topic distribution, $\vec{\vartheta}_x$, becomes $q(x, z \oplus y)$, and it can be seen that words and categories influence the association of topics to authors directly. By setting the number of label samples per document, J_m, we can control their influence on the topics.
5. *Prediction*: Model properties are derived from Fig. 2 and shown in Fig. 3, with full conditionals (7a–b) and likelihoods (7c–d), as well as recommendation tasks in (7e–f). The Gibbs sampler in (7f) makes the node $\vec{a}_m$ unsupervised and starts with a fair distribution $a_{qx} = 1/A$, updating for $\vec{w}_q$.

ETT1 has the disadvantage that it only supports a single label per document and that it does not directly model the dependence between words and labels (see (7c)). Therefore, we iterate the structure:



$$p(\{x_{mn}{=}x, z^1_{mn}{=}k \,|\, w_{mn}{=}w, \{\vec{x}, \vec{y}, \vec{z}^1, \vec{w}\}_{-mn}, \vec{z}^2) \propto a_{m,x} q(x,k) q(k, w \oplus \tilde{w}) \tag{8a}$$

$$p(\{y_{mn}{=}y, z^2_{mn}{=}k \,|\, w_{mn}{=}w, \{\vec{c}, \vec{x}, \vec{z}^2, \vec{w}\}_{-mn}, \vec{z}^1) \propto c_{m,y} q(y,k) q(k, \tilde{w} \oplus w) \tag{8b}$$

$$p(w_{mn}|\vec{a}_m, \vec{c}_m, \Theta) = \textstyle\sum_z (\sum_x a_{m,x} \vartheta_{x,z} + \sum_y c_{m,y} \zeta_{y,z})/2 \cdot \varphi_{z,w} \tag{8c}$$

$$p(a|c_q, \Theta) = \textstyle\sum_z p(a|z,\Theta) p(z|c_q,\Theta) \propto \sum_z \vartheta_{a,z} p(a|\Theta) \zeta_{c_q,z}\,, \quad p(a|\Theta) \propto \textstyle\sum_{mn} \delta(x_{mn}{-}a) \tag{8d}$$

$$p(a|\vec{w}_q, \Theta) \propto \textstyle\sum_n \delta(x_{qn}{-}a)\,,\ x_{qn} \sim \text{Gibbs (8a,b) with } a_{m,x} = q(m, x|\eta_a) \tag{8e}$$

Fig. 4. Expert–Tag–Topic model, iteration 2

6. *Iteration*: To allow multiple labels, we may actually use the same structure as the ATM for labels and merge both with a C2 or C3 structure. Here label and author-generated topics merge. The "ETT2" model is shown in Fig. 4 with properties (8a–e). For unseen documents with unknown labels or authors, the sampler is run using (8e) analogous to (7f), using unsupervised $\vec{a}_q$ and $\vec{c}_q$ with priors $\eta_.$.

Beyond these variants of an Expert–Tag–Topic model, there are various alternatives, for instance, instead of the central E2 and C3 structures we may use the E3 and C2 ones and may obtain a more straight-forward recommendation rule than (8d) in Fig. 4. As has been discussed above, the basic approach of model design needs to be complemented with guidelines for the selection of model structures given a task and dataset at hand, so the best structure types may be identified from the outset.

Furthermore, actual model performance is likely to depend on the finer details of co-occurrence structure in the data and the questions asked about them, and with these details the mileage of different models may vary. Looking at the summing structure in likelihoods (7c) and (8c) indicates that at least the models are in principle able to reach the likelihood of LDA, while full conditionals (7a–b) and (8a–b) seem to create the right "gradient" in this direction, increasing co-occurrences in hidden nodes where they are assumed in the data.

5 Empirical Analysis

Testing a complete framework of models like the one in question is the necessary step to prove the applicability of the design method. However, this larger task is ongoing work. In this paper, we limit ourselves to a verification of the results of the design approach taken and performed a proof-of-concept test of the ETT models.

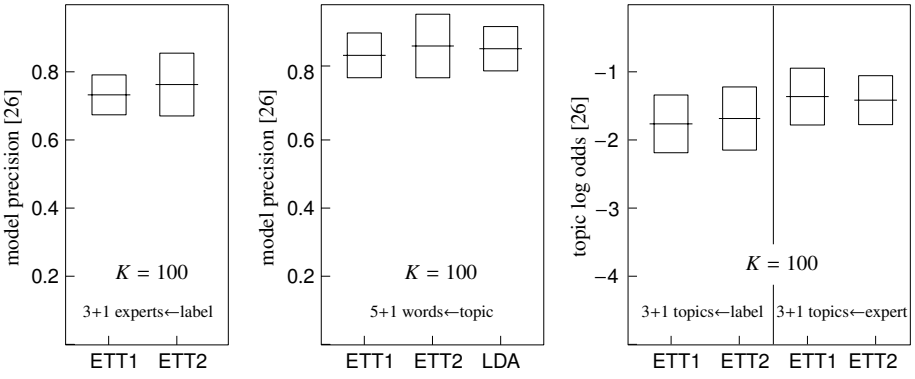


Fig. 5. Semantic coherence of ETT output

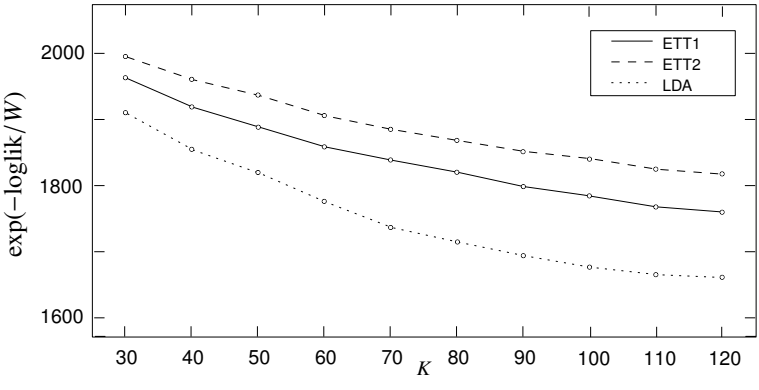


Fig. 6. ETT likelihood against baseline

As data set, we consider the NIPS corpus commonly used in topic model research, with $M = 1740$ documents, $V = 13649$ unique terms, $W = 2301375$ words, $A = 2037$ authors with $W_A = 3990$ authorship relations, and $C = 50$ categories with $M_C = 1254$ labelled documents (conference tracks and manual labels). For both models ETT1 and ETT2, Gibbs sampling was run over a range of K and the tasks performed on the trained parameters for 20 topics, 10 labels and 10 experts.

Precision of recommendations was measured by human judgements of five expert voters in the spirit of a topic-coherence experiment [26]: Voters are presented with groups of (a) 6 topic words and (b) 4 document topics and asked to detect the least consistent item. In every question, items presented have high probability according to the model, except for an unlikely "intrusion" item that participants may easier identify in semantically coherent groups. For our scenario, beside topic coherence we adapted the experiment to test associations of experts with labels (showing titles and frequent words of experts' papers), as well as topics for both experts and labels to check topic coherence. We measured the model precision and topic log odds from [26]. Results are

shown in Fig. 5, and it generally can be seen that both ETT models produce coherent output with low intrusion votings (higher values better), validating the model.

We also tested the log likelihood of held-out documents (prediction of second half of test documents, as proposed by [2]) as a "control metric" and compared against the baseline model LDA, and the result is given in Fig. 6. The ETT models are slightly inferior, but this is in line with results of [26] that report some deviation of human perception of topic coherence from model generalizability measured by likelihood. Our model creates *different* topics; it is not designed to compete with LDA that can freely adapt to the data available, taking constraints from author and label contexts. Notably, for the scenario considered here, the proposed NoMM structure C3 turns out as a viable alternative with low model complexity.

6 Conclusions and Future Work

In this article, we have shown how mixed-membership models can be separated into sub-structures, and how the sub-structures may be used as a "library" to create models according to a straight-forward design workflow. The method proposed is based on mapping qualitative assumptions to model structures and allows to stay aware of quantities like full conditional distributions of Gibbs samplers and data likelihood, important predictors of the model performance to be expected.

We have applied the design method successfully to an example scenario of expertise finding from labeled documents, but more work needs to be done in order to refine and validate the method itself. Ongoing work [27] applies the method to other scenarios and looks into refined mapping rules between data properties and model structures in order to obtain clearer modeling guidelines. A more extensive validation will be based on synthetic data with controlled properties that benchmark the different model structures, and a special aspect to look at in this context is the relation between model structures and higher-order co-occurrences in multimodal data, analogous to language data [28].

Finally, we will study how "networks of mixed membership", the model representation used here, may be used as a generalized representation of the finite models discussed here and non-parametric variants with Dirichlet or Pitman-Yor process priors [29].

References

1. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
2. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. In: 20th Conference on Uncertainty in Artificial Intelligence (2004)
3. Heinrich, G.: A generic approach to topic models. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5781, pp. 517–532. Springer, Heidelberg (2009)
4. Heinrich, G., Goesele, M.: Variational bayes for generic topic models. In: Mertsching, B., Hund, M., Aziz, Z. (eds.) KI 2009. LNCS, vol. 5803, pp. 161–168. Springer, Heidelberg (2009)
5. Pearl, J.: Bayesian networks: A model of self-activated memory for evidential reasoning. In: Proc. 7th Conf. of the Cognitive Science Society, pp. 329–334 (1985)

6. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences 101, 5228–5235 (2004)
7. Liu, J.: The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problems. Journal of the American Statistical Association 89(427), 958–966 (1994)
8. Andrews, G.E., Askey, R., Roy, R.: Special functions. Cambridge University Press, Cambridge (1999)
9. Li, W., McCallum, A.: Pachinko allocation: DAG-structured mixture models of topic correlations. In: ICML 2006: Proceedings of the 23rd International Conference on Machine Learning, pp. 577–584. ACM, New York (2006)
10. Shafiei, M.M., Milios, E.E.: Latent Dirichlet co-clustering. In: ICDM 2006: Proceedings of the Sixth International Conference on Data Mining, pp. 542–551. IEEE Computer Society, Washington, DC, USA (2006)
11. Blei, D., Lafferty, J.: A correlated topic model of Science. Annals of Applied Statistics 1, 17–35 (2007)
12. Wallach, H.M.: Structured Topic Models for Language. PhD thesis, University of Cambridge (2008)
13. Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D., Jordan, M.: Matching words and pictures. JMLR – Special Issue on Machine Learning Methods for Text and Images 3, 1107–1136 (2003)
14. McLachlan, G., Peel, D.: Finite Mixture Models. Wiley, Chichester (2000)
15. Blei, D., McAuliffe, J.: Supervised topic models. In: Advances in Neural Information Processing Systems (2007)
16. Ramage, D., Heymann, P., Manning, C.D., Garcia-Molina, H.: Clustering the tagged web. In: Proc. WSDM (2009)
17. Xu, Z., Tresp, V., Yu, K., Kriegel, H.P.: Infinite hidden relational models. In: Proc. 22nd Conference in Uncertainty in Artificial Intelligence UAI (2006)
18. Erosheva, E., Fienberg, S., Lafferty, J.: Mixed membership models of scientific publications. PNAS 101, 5220–5227 (2004)
19. Li, W., Blei, D., McCallum, A.: Mixtures of hierarchical topics with pachinko allocation. In: International Conference on Machine Learning (2007)
20. Porteous, I., Bart, E., Welling, M.: Multi-HDP: A non-parametric Bayesian model for tensor factorization. In: Proc. AAAI (2008)
21. Titov, I., McDonald, R.: Modeling online reviews with multi-grain topic models. In: Proc. 17th International World Wide Web Conference (WWW 2008), Beijing, China (2008)
22. Newman, D., Chemudugunta, C., Smyth, P.: Statistical entity-topic models. In: KDD 2006: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 680–686. ACM, New York (2006)
23. Sinkkonen, J., Parkkinen, J., Aukia, J., Kaski, S.: A simple infinite topic mixture for rich graphs and relational data. In: Proc. NIPS Workshop on Analyzing Graphs: Theory and Applications (2008)
24. Chang, J., Blei, D.M.: Relational topic models for document networks. In: AISTATS (2009)
25. Cao, L., Fei-Fei, L.: Spatially coherent latent topic model for concurrent object segmentation and classification. In: Proc. ICCV (2007)
26. Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., Blei, D.: Reading tea leaves: How humans interpret topic models. In: Proc. Neural Information Processing Systems, NIPS (2009)
27. Heinrich, G.: Typology of mixed-membership models: Applications to community data. Technical note TN2011/2, arbylon.net (2011)
28. Heyer, G., Bordag, S.: A Structuralist Framework for Quantitative Linguistics. In: Aspects of Automatic Text Analysis. Studies in Fuzziness and Soft Computing. Springer, Heidelberg (2007)

29. Teh, Y.W., Jordan, M.I.: Hierarchical Bayesian nonparametric models with applications. In: Hjort, N., Holmes, C., Müller, P., Walker, S. (eds.) To appear in Bayesian Nonparametrics: Principles and Practice. Cambridge University Press, Cambridge (2009)

A NoMM Gibbs Sampling

To keep this paper self-contained, Gibbs sampling in NoMMs is outlined for the case of multinomial levels with Dirichlet priors.

NoMMs estimate latent variables using two assumptions: (1) Models consist of a set of discrete mixtures whose multinomial parameters are generated from conjugate Dirichlet distributions. Each discrete mixture is governed by the following generative process (omitting level superscripts ℓ):

$$x_i \sim \mathrm{Mult}(x_i|\vec{\vartheta}_k) \qquad \vec{\vartheta}_k \sim \mathrm{Dir}(\vec{\vartheta}_k|\vec{\alpha}_j) \tag{9}$$

where x_i is one discrete data point (token), latent or observed, $\vec{\vartheta}_k$ is a vector of multinomial parameters with k the mixture component index and $\vec{\alpha}_j$ the parameter of the Dirichlet prior distribution (scalar or vector). These discrete mixtures are (2) coupled by discrete variables x_i to choose a component k:

$$k = f_k(\mathrm{parents}(x_i), i) \; . \tag{10}$$

This results in dependencies between the x_i (and $\vec{\vartheta}_k$) of different levels, which allows modelling of complex co-occurrences in the data. Further, components may be grouped by drawing them from different hyperparameters $\vec{\alpha}_j$, where the group indicator j may be a function of values known at the time of generating $\vec{\vartheta}_k$ due to (9):

$$j = f_j(\mathrm{known\_parents}(x_i), i) \; . \tag{11}$$

The conjugacy between the multinomial and Dirichlet distributions of model levels leads to a simple complete-data likelihood:

$$p(X, \Theta|A) = \prod_\ell \prod_i \mathrm{Mult}(x_i^\ell|\vec{\vartheta}^\ell, k^\ell) \prod_k \mathrm{Dir}(\vec{\vartheta}_k^\ell|\vec{\alpha}_j^\ell) \tag{12}$$

$$= \prod_\ell \left[\prod_k \frac{\mathrm{B}(\vec{n}_k + \vec{\alpha}_j)}{\mathrm{B}(\vec{\alpha}_j)} \mathrm{Dir}(\vec{\vartheta}_k|\vec{n}_k + \vec{\alpha}_j) \right]^\ell \tag{13}$$

where brackets $[\cdot]^\ell$ enclose a particular level ℓ.

Gibbs full conditionals are derived for groups of dependent hidden edges, $H^d \subset X$ (with dependent tokens $h_i^d \in H^d$) and their "surrounding" edges S^d (with $s_i^d \in S^d$) considered observed. We also define the set of all tokens co-located with a particular observation, $x_i^d = \{h_i^d, s_i^d\}$ where i (actually i^d) is the sequence of token indices for group d. For each of the dependency groups thus defined, a full conditional is created, using (13) with Θ integrated out:

$$
\begin{aligned}
p(h_i^d \,|\, X \backslash h_i^d, A) &= \frac{p(h_i^d, s_i^d \,|\, X \backslash \{h_i^d, s_i^d\}, A)}{p(s_i^d \,|\, X \backslash \{h_i^d, s_i^d\}, A)} \\
&\propto p(x_i^d \,|\, X \backslash x_i^d, A) = \frac{p(X \,|\, A)}{p(X \backslash x_i^d \,|\, A)} \\
&= \prod_{\ell} \left[\prod_{k} \frac{\mathrm{B}(\vec{n}_k + \vec{\alpha}_j)}{\mathrm{B}(\vec{n}_k \backslash x_i^d + \vec{\alpha}_j)} \right]^{\ell} \\
&\propto \prod_{\ell \in L(H^d)} \left[\frac{\mathrm{B}(\vec{n}_k + \vec{\alpha}_j)}{\mathrm{B}(\vec{n}_k \backslash x_i^d + \vec{\alpha}_j)} \right]^{\ell}
\end{aligned}
\tag{14}
$$

where $L(H^d)$ is the set of all levels ℓ whose variables interact with the edges in the set H^d. For a single hidden variable set H^d, this leads to (11). Note that $\backslash x_i^d \equiv \backslash x_{i^d}^d$ excludes more than a single token from a particular edge if a token with index i^d at node input corresponds to multiple tokens $c \in i^\ell$ at its output, which leads more complex terms than (12). This occurs in Fig. 2 for structures E2 and when data aggregations are explicitly modelled using E1b structures (e.g. [10]). In (13), this case of "sub-tokens" is excluded for simplicity, and a complete formulation may redefine the parameter ϑ_{kt} as a product of sub-token likelihoods: $\vartheta_{kt} = \prod_{c \in i^\ell} \vartheta_{kt_c}$. This expands to a hierarchy of products for recursive sub-sequences.

ShiftTree: An Interpretable Model-Based Approach for Time Series Classification

Balázs Hidasi and Csaba Gáspár-Papanek

Budapest University of Technology and Economics,
Department of Telecommunication and Media Informatics,
{hidasi,gaspar}@tmit.bme.hu

Abstract. Efficient algorithms of time series data mining have the common denominator of utilizing the special time structure of the attributes of time series. To accommodate the information of time dimension into the process, we propose a novel instance-level cursor based indexing technique, which is combined with a decision tree algorithm. This is beneficial for several reasons: (a) it is insensitive to the time level noise (for example rendering, time shifting), (b) its working method can be interpreted, making the explanation of the classification process more understandable, and (c) it can manage time series of different length. The implemented algorithm named ShiftTree is compared to the well-known instance-based time series classifier 1-NN using different distance metrics, used over all 20 datasets of a public benchmark time series database and two more public time series datasets. On these benchmark datasets, our experiments show that the new model-based algorithm has an average accuracy slightly better than the most efficient instance-based methods, and there are multiple datasets where our model-based classifier exceeds the accuracy of instance-based methods. We also evaluated our algorithm via blind testing on the 20 datasets of the SIGKDD 2007 Time Series Classification Challenge. To improve the model accuracy and to avoid model overfitting, we provide forest methods as well.

Keywords: model-based time series classification, decision trees, forest building methods.

1 Introduction

With the spread of automatic data collection systems, the role of time series has been increasing in business intelligence applications in the domains of entertainment, industry and of mobile devices. Even though the traditional source of time series databases is the financial sector, due to the decrease in the pricing of sensors, more and more time series data are collected from everyday electrical devices.

For example, most new cellular phones and laptops have a gyroscope for the collection of acceleration data. By processing these time series data, hand gesture controlled interfaces can be built into many applications. There seems to have been an increase in the number of time series based applications on the end-user level. Time series data can also be found in the fields of medicine and

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 48–64, 2011.

biology (e.g.: the ECG(electrocardiographic signal)), finance, system monitoring and logistics.

Naturally, supervised and unsupervised learning tasks also appear connected to time series data. These data mining tasks can be organized into the following categories:

1. Data mining of single time series
   (a) Next value prediction in time series (e.g. Stock market prediction [3])
   (b) Clustering of segments of the time series (e.g. Time series subsequence clustering)
   (c) Classification of segments of the time series (e.g. Hand gesture recognition in accelerator data [15])
   (d) Motif (similar subsequences) discovery in a longer time series [14]
2. Data mining of multiple time series
   (a) Clustering of time series (e.g. Segmentation of customers of an electricity provider by clustering the time series of their charging)
   (b) Classification of time series (e.g. Analyzing heart function by classification of ECG signals [2])

The complexity of this hierarchy can be reduced if we consider the fact that Points 1.b and 1.c can be incorporated into Case 2. by the segmentation of the original time series.

The reason for the difficulty of these tasks is rooted in the multi-dimensional problem space and the special connection between the attributes (element or values of time series): the sequence of attributes (elements) carries information about the source entity. In the case of traditional vector-based data representation, there is no information in the order of attributes, but time series elements, which are close to each other, have special connection through the dimension of time. For example, if the values are shifted in a time series by one position (for example, value of attributes i is replaced by attributes i-1), then the classification label or cluster ID of the time series will probably stay the same. The effective algorithms of time series data mining typically have some additional aspect to handle the effect of this time-dimension structure. Our new method is capable of considering time level aspects of time series.

Our approach is a novel model-based classification method labeling different time series by learning from the database with unknown labels. We named this algorithm ShiftTree. The beneficial properties of the method are the following:

- accuracy level similar to other techniques
- interpretable model
- capable of handling datasets of time series with different lengths
- preprocessing not necessary
- expert knowledge can be built into the modeling process.
- correspondences coded in time dimension can be interpreted

The rest of this paper is organized as follows: Section 2 reviews the time series classification techniques, Section 4 presents the concept of our novel approach

ShiftTree, whereas its formal definition is described in Section 5. After the Section about interpretability, the Section 6 provides two other techniques to improve the accuracy. Section 7 summarizes the numerical results, finally, Section 8 sums up our experiments.

2 Related Works

Time series specific classification algorithms usually belong to two categories: instance-based (memory-based) learning methods form hypotheses directly from the training instances themselves, whereas model-based learning methods create general coherence by describing the implicit information of training data.

The key aspects of instance-based time series classifiers (e.g. k-nearest neighbor algorithm and its variations) are the representation methods and the (de)similarity measures. Time series representation techniques deal with the transformation of the high-dimensional time series data to an other feature space. The well-known representation methods are the Discrete Fourier Transformation (DFT) [8] , Singular Value Decomposition (SVD) [8] , Discrete Wavelet Transformation (DWT) [4] etc.

Their main functions are noise filtering and feature extraction. The similarity measures have more connections to the special attribute structure of time series, some of them are called elastic measures because they tolerate partial shifting or spreading of the time series values. Dynamic Time Warping (DTW) [11] and the edit distance based methods (Longest Common SubSequence(LCSS) [18] , Edit Distance on Real Sequence (EDR) [6] and Edit Distance with Penalty (EDP) [5]) are very efficient elastic similarity measures.

Typically, instance-based methods in time series classification provide efficient and accurate solutions [7] , but the selection of the appropriate representation method and the similarity measure require difficult cross-validation steps, more running time and expert knowledge. Extensive experimental comparision of representation and simirality measure can be read in [7].

Most model-based methods include some submethods to generate or predict the time series. For example, a Hidden Markov Model (HMM) can be built on time series with the same label, thus a time series in the test set is associated with the class of which HMM has the highest probability to generate the given time series.[17]. Similarly to this method, an other time series prediction method can be used in a classification algorithm, in which case the higher accuracy of the prediction method determines the labeling of the predicted time series. One member of the most popular and efficient time series prediction method family is the recurrent neural networks [10]. The classifiers based on these neural networks are accurate, however, their models are non-interpretable.

Typically, the instance-based method can not handle time series with different lengths, they require time series with equal lengths, whereas the prediction-based solutions can handle difference in the length of time series as well.

3 Classification of Time Series

3.1 Problem Definition

Time series Θ is a structured data, a finite vector of time value and observation vector pairs ($\Theta = \{< t_i, \boldsymbol{x_i} >\}_{i=1}^{T}$ where $\boldsymbol{x_i} =< x_i^1, x_i^2, \ldots, x_i^m >, x_i^j \in \Re$). The vector is ordered by the time parameter of its elements ($t_i \leq t_{i+1}$). In this paper we concentrate on equally sampled time series where $t_{i+1} - t_i$ equals $t_{j+1} - t_j$, and we assume that t_i equals i, so we can simplify Θ to $\{\boldsymbol{x_i}\}_{i=1}^{T}$ (i.e. a vector of observation vectors). Although the ShiftTree is also capable of classifying time series with multiple observations (i.e. multinomial time series), we concentrate on a simpler task in this paper: the $\boldsymbol{x_i}$ observation vector is replaced by x_i observation scalar. This type of structured data is also called value series in the literature. In the rest of this paper time series Θ refers to a series of x_i values.

In the classification task, we are given a training set of time series with class labels ($TR = \{< \Theta_n, L_n >\}_{n=1}^{N_{TR}}$) and a set of time series with unknown class labels. The task is to determine the value of the class labels of the elements of the latter set. The class labels get their values from a finite (and often small) set of values ($L_n \in CL = \{l_1, l_2, ..., l_{N_C}\}$). For the evaluation and comparison of different classifiers, a test set is used ($TE = \{< \Theta_n, L_n >\}_{n=1}^{N_{TE}}$). The TR and TE sets have no common elements.

There are many metrics for evaluating classifiers. In this paper we use accuracy. The classifier assigns a predicted class label $\hat{L_n}$ to the n^{th} series of the TE set. We define $\#hits$ as the number of correctly predicted class labels and the accuracy of the classifier as $Accuracy = \frac{\#hits}{N_{TE}} = \frac{\sum_{n=1}^{N_{TE}} Ind\{L_n = \hat{L_n}\}}{N_{TE}}$

3.2 Notation

- $TR \rightarrow$ The training set.
  - $N_{TR} \rightarrow$ The number of time series in the training set.
  - $TR[n] =< \Theta_n, L_n > \rightarrow$ The n^{th} element of the training set, a time series and class label pair.
  - $\Theta_n \rightarrow$ The n^{th} series in the training set.
  - $L_n \rightarrow$ The class label of the n^{th} series in the training set.
- $TE \rightarrow$ The test set. The meaning of N_{TE}, $TE[n]$, Θ_n^{te} and L_n^{te} are similar to N_{TR}, $TR[n]$, Θ_n and L_n. (The n^{th} series of the TR and TE sets are distinguished by the superscript ^{te}.)
- $CL \rightarrow$ The set of the possible class labels $\{l_1, l_2, ..., l_{N_C}\}$.
  - $N_C \rightarrow$ The number of different class labels.
- $\Theta \rightarrow$ A time series.
  - $\Theta[i] \rightarrow$ The i^{th} observation value of the time series Θ (i.e. x_i).
  - $T \rightarrow$ The length of the time series.

4 Concept

In this paper we propose a novel decision tree based algorithm called ShiftTree. In a node of an ordinary decision tree, the data set splitting criteria belongs to only a certain attribute x_i, but in the case of time series, adequate information is usually not in the same attribute x_i for each time series, as it may be found in an different attribute position for each time series. For example, the global maximum of time series would be an efficient splitting attribute, but their values can not be assigned to an exact position i in time series, to a certain attributes x_i, so the approach of vector based attribute representation is not adequate in this case.

In order to handle these problems, we assign a cursor (or eye) - denoted C - to every time series. The task of the cursor is to appoint an element of time series, and it can be interpreted as a position of its time series. This cursor can move back and forward on the time axis of the time series throughout the duration of our method. Initially, the cursors are set to the first position/attribute of the time series (It's assigned the attribute x_1, its value is 1). In our algorithm, every node of the decision tree has an operation of cursor, for example the cursor has to move to the next local maximum of time series. The result of this operation would be different for different time series, so this method has the possibility of implementing a time-elastic handling of time dimension. Attributes are computed dynamically using the position of the cursor, the value of the time series in that position and the surrounding values.

Every node of ShiftTree has an operation of computing the attribute, for example the attribute is the average of the values in the surroundings of the cursor with a radius of 5. In this way, each branch of the ShiftTree gives an interpretable description of the time series. An other important advantage of this approach is that the expert of the application field can define suitable operations to create a more accurate model for a specific problem.

The training of the novel decision tree model is based on selecting the appropriate operators (moving of cursor, attribute calculator) for each node. Our proposed training method is described in Section 5. The accuracy of the model depends on the set of usable operations from which a node can choose an appropriate one. One of our main goals was to create a general algorithm which is applicable to different fields, which we will display in Section 7, by using a basic set of operations to show that the accuracy of the models is satisfying on severely different time series classification problems. The accuracy can be further improved by using forest building methods. In section 6 we present two forest building methods based on boosting and cross-validation.

5 The ShiftTree Algorithm

5.1 The Structure of a ShiftTree Node

The main structure of our proposed algorithm is the ShiftTree, which is similar to the structure of decision tree algorithms: it is a binary tree with a root node,

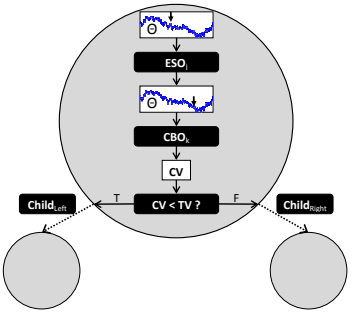


Fig. 1. Structure of a ShiftTree node

the leaf node contains the classification labels and decision points are associated with the not leaf nodes. As we mentioned above every node of the ShiftTree contains two operators: the first one describes how to move the cursor, the second one describes how to compute a dynamic attribute. The family of the first operator type is called EyeShifter operator (ESO) the second group is called ConditionBuilder operator (CBO).

Each node of the ShiftTree can be represented by the following structure of six elements $< ESO_j, CBO_k, TV, PLabel, Ch_L, Ch_R >$. ESO_j is an EyeShifter operator selected from a predefined set of ESOs ($j \in [1..N_{ESO}]$). An EyeShifter Operator ESO_j describes a shifting mode of a cursor on the given time series. It is important to understand that an ESO_j can shift the cursor on different time series to different positions. That is why the method can handle time series of different lengths in one classification task. CBO_k is a ConditionBuilder operator selected from a predefined set of CBOs ($k \in [1..N_{CBO}]$). CBO_k generates a dynamic attribute called Calculated Value (CV) using the position of the cursor C, the value of the time series in that position ($\Theta[C]$) and the nearby values. Ch_L and Ch_R are pointers to the left and right subtrees of the current node. If the node is a leaf then these two values are null. TV is called the threshold value. If the corresponding attribute of the time series is smaller than TV then the branch pointed by Ch_L will be the next one, in other cases the branch pointed by Ch_R will process the time series. The function $PLabel$ describes the labeling information in the node, $PLabel(l_i)$ returns with the confidence (probability) of the label l_i in the given node. The structure of a node is shown in Figure 1.

We present some simple operator examples for both ESO and CBO. The current position of the cursor is denoted by C, C_{new} is the new position of the cursor after applying the ESO, C_{prev} is the previous position of the cursor. The parameters of the operators are predefined, they are not changing during the learning process. We will show in Section 7 that a ShiftTree can be accurate using only this simple operator set.

Operator Examples

- ESONext(ΔT) $\rightarrow C_{new} = min(C+\Delta T, T)$. Similar operator: ESOPrev(ΔT).
- ESONextMax(X) $\rightarrow C_{new} = (i|\Theta[i] > max\{\Theta[i \pm 1]\}, C < i, \sum_{k=C+1}^{i-1} I_{\Theta[k]>max\{\Theta[k\pm1]\}} = X-1)$. Similar operator: ESOPrevMax(X), ESONextMin(X), ESOPrevMin(X).
- ESOMax(global) $\rightarrow C_{new} = argmax_i(\Theta[i])$. Similar operator: ESOMin(global).
- ESOMax(sofar) $\rightarrow C_{new} = argmax_{i=1\ldots C}(\Theta[i])$. Similar operator: ESOMin(sofar).
- ESOClosestMax $\rightarrow C_{new} = min_{|C-i|}(i|\Theta[i] > \Theta[i-1], \Theta[i] \geq \Theta[i+1])$ Similar operator: ESOClosestMin.
- ESOGreaterMax $\rightarrow C_{new} = argmax(\Theta[ESONextMax(1)], \Theta[ESOPrevMax(1)])$ Similar operators: ESOGreaterMin, ESOLesserMax, ESOLesserMin.
- ESOMaxInNextInterval(ΔT) $\rightarrow C_{new} = argmax_{i=0\ldots\Delta T}(\Theta[C+i])$. Similar operators: ESOMaxInPrevInterval(ΔT), ESOMinInNextInterval(ΔT), ESOMinInPrevInterval(ΔT).
- ComplexESO $\rightarrow$ This operator is a vector of two or more ESOs. It moves the cursor by its first ESO then by its second ESO and so on.
- CBOSimple $\rightarrow CV = \Theta[C]$
- CBONormal(μ,σ,X) $\rightarrow CV = average\{\exp^{-\frac{\mu^2}{2\sigma^2}}\Theta[C], \exp^{-\frac{(|i|-\mu)^2}{2\sigma^2}}\Theta[C \pm i]|i = 1\ldots X\}$
- CBOExp(λ,X) $\rightarrow CV = average\{\lambda\Theta[C], \exp^{-\lambda|i|}\Theta[C \pm i]|i = 1\ldots X\}$
- CBOLinear(X) $\rightarrow CV = average\{\Theta[C], \frac{1}{|i|}\Theta[C \pm i]|i = 1\ldots X\}$
- CBOAVG(X) $\rightarrow CV = average\{\Theta[C], \Theta[C \pm i]|i = 1\ldots X\}$
- CBODeltaT(norm/abs) $\rightarrow CV = C - C_{prev}$ or $CV = |C - C_{prev}|$
- CBOTimeSensitive(norm/abs) $\rightarrow CV = \frac{\Theta[C]}{C-C_{prev}}$ or $CV = \frac{\Theta[C]}{|C-C_{prev}|}$
- CBO[Average/Variance](sofar/delta) $\rightarrow$ Returns the average/variance of the values $\{\Theta[1], ..., \Theta[C]\}$ or $\{\Theta[C_{prev}], ..., \Theta[C]\}$
- CBO[Max/Min][AVG/VAR/Count](sofar/delta) $\rightarrow$ Returns the average/variance/number of the local maximums/minimums in the subseries of $\{\Theta[1], ..., \Theta[C]\}$ or $\{\Theta[C_{prev}], ..., \Theta[C]\}$.
- CBOMedian(B, F) $\rightarrow CV = median\{\Theta[C-B], \Theta[C-B+1], ..., \Theta[C], ..., \Theta[C+F-1]\Theta[C+F]\}$

5.2 Classification Process

The ShiftTree's classification process for time series Θ can be written by the next recursive process (see Algorithm 5.1). The input of the first call has to be a ShiftTree represented by its root node R and the unlabeled time series Θ and the initial cursor position ($C = 0$).

The function $ShiftCursor(ESO_j, \Theta, C)$ shifts the cursor of time series from position C to a new one by applying EyeShifter operator ESO_j, the function $CalculateValue(CBO_k, \Theta, C)$ calculates a value over time series Θ by using ConditionBuilder operator CBO_k and the cursor position C.

Algorithm 5.1. Labeling process of the ShiftTree

Input: node R, time series Θ, cursor C
Output: label $L \in [l_1, ..., l_{N_L}]$ for time series Θ
procedure ShiftTreeLabel(R, Θ, C)

```
1: R →< ESO_j, CBO_k, TV, PLabel, Ch_L, Ch_R >
2: if R is not a leaf then
3:     C_new ← ShiftCursor(ESO_j, Θ, C)
4:     CV ← CalculateValue(CBO_k, Θ, C_new)
5:     if CV < TV then
6:         L ← ShiftTreeLabel(Ch_L, Θ, C_new)
7:     else
8:         L ← ShiftTreeLabel(Ch_R, Θ, C_new)
9:     end if
10: else
11:    L ← argmax_{l_i} PLabel(l_i), l_i ∈ [l_1, ..., l_{N_L}]
12: end if
13: return L
```

end procedure

5.3 Training Process

The learning process of the ShiftTree is more complicated (see Algorithm 5.2). The process is defined by the generation method of only one ShiftTree node, because the training method can be defined as a recursive algorithm. In this case the input is a training set $TR = \{< \Theta_n, L_n >\}_{n=1}^{N_{TR}}$. The output of process is a subtree of the ShiftTree represented by its root node R. The process tries to find an accurate ESO_j, CBO_k and TV setting, because this triple determines a splitting criteria in a given node. The algorithm selects the best splitting criteria by minimizing the entropy of the child nodes. Note that this is the same as maximizing the information gain of the splitting. The entropy is defined as follows:

$$Ent(TR_L, TR_R) = - \sum_{X \in [L,R]} \frac{N_X}{N} \sum_{i=1}^{N_C} (P_{Xi} * \log_2 P_{Xi}) \quad (1)$$

TR_L and TR_R are the two sets of time series label pairs. N , N_L and N_R are the number of time series in $TR_L \bigcup TR_R$, TR_L and TR_R. P_{Li} and P_{Ri} are the relative frequency of the label l_i in TR_L and TR_R. N_C is the number of class labels.

The function $StoppingCriteria(PLabels)$ return true if $PLabels(l_i) = 1$ for a class label value $l_i \in CL$. We experimented with other stopping criteria but this one gave the best results on the benchmark datasets. If the node is not a leaf, every ESO_j CBO_k pairs are examined by the training algorithm. $ShiftCursor(ESO_j, \Theta, C)$ and $CalculateValue(CBO_k, \Theta, C)$ are the same as they were in algorithm 5.1. When the CVs are calculated for all time series in TR, every sensible threshold value is examined. The easiest way to do this is to sort the CVs and set TV to be the mean of every two adjacent CVs

Algorithm 5.2. Recursive learning method of ShiftTree

Input: a set of labeled time series $TR = \{< \Theta_n, L_n >\}_{n=1}^{N_{TR}}$ and their cursors $\{C_n\}_{n=1}^{N_{TR}}$
Output: Node R that represents the newly created subtree of the ShiftTree
procedure BUILDSHIFTTREE(TR,$\{C_n\}_{n=1}^{N_{TR}}$)

New node R
for all $l_i \in CL$ **do**
  $PLabels(l_i) \leftarrow \frac{|\{n|L_n=l_i\}|}{N_{TR}}$
end for
if STOPPINGCRITERIA($PLabels$) = true **then**
  $R \leftarrow leaf$
else
  for all $ESO_j \in ESO$ **do**
    for all $CBO_k \in CBO$ **do**
      for $n = 1..N_{TR}$ **do**
        $C_{new}^{j,n} \leftarrow$ SHIFTCURSOR(ESO_j,Θ_n,C_n)
        $CV^{j,k,n} \leftarrow$ CALCULATEVALUE(CBO_k,Θ_n,$C_{new}^{j,n}$)
      end for
      $[CV^{j,k,1}, CV^{j,k,2}, \ldots, CV^{j,k,N_{TR}}] \leftarrow$ SORT($[CV_1^{j,k}, CV_2^{j,k}, \ldots, CV_{N_{TR}}^{j,k}]$)
      for $m = 1..N_{TR} - 1$ **do**
        $TV^{j,k,m} \leftarrow (CV_m^{j,k} + CV_{m+1}^{j,k})/2$
        $TR_L^{j,k,m} \leftarrow \{\Theta_n | CV^{j,k,n} < TV\}$
        $TR_R^{j,k,m} \leftarrow \{\Theta_n | CV^{j,k,n} \geq TV\}$
        $E^{j,k,m} \leftarrow$ ENT($TR_L^{j,k,n}, TR_R^{j,k,n}$)
      end for
    end for
  end for
  $< j'_q, k'_q, m'_q >_{q=1}^{Q} = \{< j,k,m > | E^{j,k,m} = min_{j,k,m} E^{j,k,m}\}$
  $j', k', m' = argmax_{j'_q,k'_q,m'_q} H_1(\{CV_n^{j'_q,k'_q}\}_{n=1}^{N_{TR}}, TV^{j'_q,k'_q,m'_q})$
  for $n = 1..N_{TR}$ **do**
    $C_n \leftarrow C_{new}^{j',n}$
  end for
  $TR_L \leftarrow TR_L^{j',k',m'}$
  $TR_R \leftarrow TR_R^{j',k',m'}$
  $Cursors_L \leftarrow \{C_n | \Theta_n \in TR_L\}$
  $Cursors_R \leftarrow \{C_n | \Theta_n \in TR_R\}$
  $Ch_L \leftarrow$ BUILDSHIFTTREE(TR_L,$Cursors_L$)
  $Ch_R \leftarrow$ BUILDSHIFTTREE(TR_R,$Cursors_R$)
  $R \leftarrow < ESO_{j'}, CBO_{k'}, TV^{j',k',m'}, PLabels, Ch_L, Ch_R >$
end if
return R
end procedure

(line 14 - 19), because by doing so we examine every possible splitting of the TR set by the current dynamical attribute. Lines 23 - 24 select the best splitting. As we mentioned above the algorithm selects the splitting which minimizes the entropy of the child nodes. In case of small training datasets, there may be several $< j, k, m >$ triplets that minimizes the expression in line 23. One should think that selecting one from equally good triplets is meaningless but our experiments have shown that the selection can significantly affect the accuracy of the model. The training set contains no trivial information to distinct these triplets properly so we rely on heuristics. We defined two similar heuristics that were based on the fact that the $CV^{j,k,n}$ values should be as far away from TV as possible. It can be assumed that a member of the test set has lower probability of ending up on the wrong side of the splitting if the CVs of the element of TR_L and TR_R are more distinct. We also had to use some kind of normalization because the CBOs might work in different range. This can be achieved in many ways, we found the following heuristic satisfying:

$$H_1(\{CV_n^{j,k}\}_{n=1}^{N_{TR}}, TV^{j,k,m}) = \frac{CV_{m+1}^{j,k} - CV_m^{j,k}}{CV_{N_{TR}}^{j,k} - CV_1^{j,k}} \tag{2}$$

If a triplet $< j, k, m >$ maximizes formula (2), then the CVs of the elements of $TR_L^{j,k,m}$ and $TR_R^{j,k,m}$ are rather distinct from each other. There may be some nodes where more than one $< j', k', m' >$ triplets minimize (1) and maximize (2), but those nodes don't seem to be significant as they usually have a TR set of only a couple of time series. In that case the first appropriate triplet is selected.

At the end of the process (lines 25 - 34) we set cursor C to its new position, split the TR set into two sets (TR_L, TR_R), create the child nodes using the same process on the elements of TR_L and TR_R. Note that Algorithm 5.2 is for demonstrative purposes only, for example collecting all possible TVs is not optimal and there are other issues one should consider when implementing this algorithm. Computational complexity may seem to be high, but a semi-optimal implementation of the ShiftTree was much faster than 1-NN using Eucledian distance or DTW.

5.4 About Interpretability

Interpretability is often underestimated but it can be of great importance in practical applications as most of the users do not trust machine learning algorithms unconditionally. If a model is interpretable, one can check if it learned an unimportant feature of the data or noise. An other advantage of interpretability - besides gaining trust of the users - is that we can learn the previously unknown properties of a problem. If a ShiftTree model is analyzed, special decision scenarios can be created by following different branches of the tree. If the ESOs and CBOs are simple interpretable operations, the experts can be understood deeper correspondences by considering the cursor scenarios.

6 Forest Methods for ShiftTree

It is a common method to create different models for a given classification problem and then combine the output of those model in order to achieve improved accuracy. The models can be the results of one or many algorithms. Building and combining only decisions trees is often called forest building. In this section we briefly introduce two forest approaches which we used to improve the accuracy of ShiftTree models.

6.1 Boosting

One of the most common methods for combining is boosting [9]. This iterative method assigns weights to the elements of the training set, trains a model and assigns a weight to the model, based on its weighted classification error. The weights assigned to the elements of the training set is also modified in a way that the weights of the correctly classified elements decrease and the weights of the rest of the elements increase. The combined output is a weighted vote on the label. The widely used AdaBoost [9] technique has a precondition that the weighted classification error of the model must be lesser than 50%. This is the same error rate as the error of random guessing on a classification problem of 2 classes. Since we tested our algorithm on some problems which have many classes (up to 50), we selected an other boosting technique that has a less strict precondition. This boosting technique is called SAMME [19] and requires an error rate lesser than $100\% - \frac{1}{N_C}$ which is the same as the error of random guessing on a problem with N_C classes. This method assigns the weight W_m to the model and increases the weight of the wrongly classified elements. The weights of the correctly classified elements are not updated but after the update the sum of all weight is normalized to 1. Like AdaBoost, this method also stops when the error of the model is 0%. We had to solve the problem that the ShiftTree often creates a model that fits to the entire training set (in other words, the model classifies every single element of the training set correctly). We experimented with many pruning techniques. Every one of them decreased the accuracy of the models on the test set. In the case of low accuracy the combined models are better than an accurate single model. We used the common chi-square post-pruning [16].

6.2 XV Method

This combination technique receives its name after cross-validation. Only a part of the training set is used for training, the other part serves as a validation set (VA) on which we measure the predicted accuracy of the model. We assign the predicted accuracy to the model as the weight of the model. The combined output is a weighted vote on the label, so this method implements a simple ensambled method over ShiftTree construction. The two parameters of this method are the iteration number M and the ratio of the sizes of the VA and (original) TR sets.

7 Numerical Results

In this section we present the results of the ShiftTree on some datasets and compare them to the accuracy of widely used instance-based methods. We examined both the basic algorithm and the forest building methods. We also did blind tests that took place in a contest environment and compared our results to the results of the participants of that competition.

7.1 Datasets and Testing Environment

We used three databases for the evaluation. The first database is one of the largest publicly available time series databases [13] often used as a benchmark database. It will be referred to as the UCR database. This database consists of 20 classification problems (datasets). Each set is originally divided into a training and a test set. We used these original splits. About half of the training sets in this database are small. While it is important to check the results of ShiftTree on these classification problems too, we do not expect high accuracy on these problems as ShiftTree is a model-based algorithm. The second database consists of the 2 datasets of the Ford Classification Challenge [1]. These datasets were originally divided into 3 sets (training, validation, test). We merged the training and validation sets into a training set by both datasets and used the test set for testing. This database will be referred to the Ford database. These two databases were used for the normal testing of our algorithm. The third database comprises the data of the SIGKDD2007 Time Series Classification Challenge [12]. This database will be referred to the TSC database. This database consists of 20 classification problems and the properties of the datasets are similar to the properties of the UCR database. We used the TSC data for the blind tests. The properties of the datasets can be seen in figure 2.

As one of our goals was to create a generally accurate algorithm, we decided to use the same operators for every problem. The description of these operators is in section 5. The parameters of the operators were also the same by all problems. The value of the parameters were determined based on the minimal and maximal length of time series of all datasets. Some operators were used more than once (with different parameterization). A total of 130 ESOs and 48 CBOs were used, so 6240 dynamic attributes were considered in each node.

7.2 Results of the Basic ShiftTree

Figure 2 shows the accuracy values for the problems of the UCR and Ford databases. The weighted accuracy is the number of correctly classified series in all test sets divided by the number of test samples (i.e. the weights are the sizes of the test sets). The accuracy of the widely used 1-NN algorithm with both Euclidian distance and DTW are also shown. We used the results reported on [13] for 1-NN.

ShiftTree has the highest overall accuracy of the 3 algorithms, but the ranking of the algorithms at different problems vary. As it is expected from a model based

ACCURACY				PROPERTIES			ACCURACY				PROPERTIES		
Name	ShiftTree	1-NN (Eucledeian)	1-NN (DTW)	|TR|	|TE|	Nc	Name	ShiftTree	1-NN (Eucledeian)	1-NN (DTW)	|TR|	|TE|	Nc
UCR							*OSULeaf*	56.20%	51.70%	59.10%	200	242	6
50Words	53.63%	63.10%	69.00%	450	455	50	*SwedishLeaf*	76.80%	78.70%	79.00%	500	625	15
Adiac	56.27%	61.10%	60.40%	390	391	37	*SyntheticControl*	92.00%	88.00%	99.30%	300	300	6
Beef	56.67%	53.30%	50.00%	30	30	5	*Trace*	100.00%	76.00%	100.00%	100	100	4
CBF	94.22%	85.20%	99.70%	30	900	3	*TwoPatterns*	99.85%	91.00%	100.00%	1000	4000	4
Coffee	82.14%	75.00%	82.10%	28	28	2	*Wafer*	99.98%	99.50%	98.00%	1000	6164	2
ECG200	100.00%	88.00%	77.00%	100	100	2	*Yoga*	85.30%	83.00%	83.60%	300	3000	2
FaceAll	65.56%	71.40%	80.80%	560	1690	14	Ford						
FaceFour	70.45%	78.40%	83.00%	24	88	4	*FordA*	93.11%	68.26%	71.29%	3601	1320	2
Fish	74.29%	78.30%	83.30%	175	175	7	*FordB*	67.04%	59.51%	65.56%	3636	810	2
GunPoint	95.33%	91.30%	90.70%	50	150	2	Weighted average accuracy						
Lightning2	72.13%	75.40%	86.90%	60	61	2	***All sets***	**89.16%**	**85.30%**	**89.09%**			
Lightning7	63.01%	57.50%	72.60%	70	73	7	***Larger sets***	**94.71%**	**89.19%**	**91.73%**			
OliveOil	66.67%	86.70%	86.70%	30	30	4	***Smaller sets***	**71.87%**	**73.18%**	**80.89%**			

Fig. 2. Results of the basic ShiftTree, 1-NN (Euclidian) and 1-NN (DTW). Also contains some basic properties of the datasets.

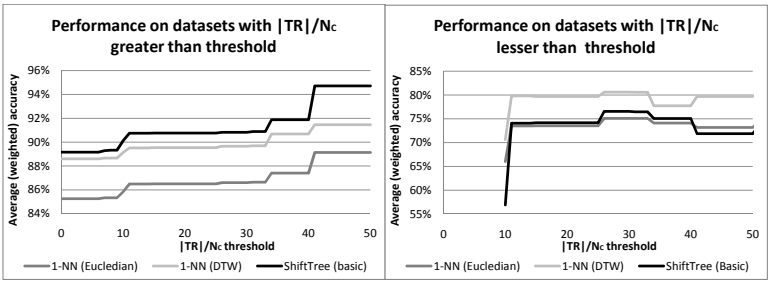


Fig. 3. Results of the basic ShiftTree, 1-NN (Euclidian) and 1-NN (DTW) on datasets with greater and lesser $\frac{|TR|}{N_C}$ values than a moving threshold

method, ShiftTree is less effective on smaller datasets. The average accuracy of the algorithms on smaller/larger datasets are shown on figure 2. We considered datasets "smaller" if the average number of instances per a class in the training set ($\frac{|TR|}{N_C}$) is lesser than than a threshold value (40). ShiftTree outperforms the neighbor based algorithms if it is provided with enough samples of every class, but loses to them if there are only a few samples available. Figure 3 shows the accuracy of all three algorithms on both "smaller" and "larger" datasets using different threshold values. At a threshold value Th the datasets of the UCR and Ford databases were divided into two groups: the ones with lesser $\frac{|TR|}{N_C}$ value than Th were considered "smaller" an the others "larger". We computed the weighted accuracy (all correctly classified test samples divided by all test samples) for both groups. By increasing the threshold, ShiftTree gains greater advantage on 1-NN using the "larger" datasets. By lowering the threshold the gap between ShiftTree and 1-NN widens on "smaller" datasets. This proves our assumption that our model based approach performs well mostly on larger datasets.

The average running time of the basic ShiftTree (training) algorithm was 6.48 seconds per dataset in case of UCR dataset collection (minimum 0,144

sec - CBF; maximum 33,99 sec - 50Words). The running times on the larger FordA and FordB datasets were 200.5 and 173.5 seconds.

7.3 Results of the Forest Methods

We made several experiments with the forest building techniques (described in 6). We found that the accuracy of boosting increases continuously as the number of iterations is increased. We finally set the number of iterations to 100 which is an acceptable trade-off between speed and accuracy. The significance level of pruning was set to 0.01% (strict pruning). Increasing number of iterations by the XV method did not really affect the accuracy above 20 so we used 20 as the M parameter. The XV method has another parameter: the size of the validation set (S). If it is set too low then the variance of the predicted accuracy values will be high and these accuracies are useless as model weights (because they are inaccurate). If we set S too high, then the ShiftTree models will be inaccurate as there are only a few samples for the training. We found 30% to be the optimal value for S. The results of both methods (using the optimal parameters) and the basic method can be seen in Figure 4. Boosting seems to be the better

Name	Basic ShiftTree	ShiftForest (boost)	ShiftForest (XV)
UCR			
50Words	53.63%	74.51%	69.89%
Adiac	56.27%	69.82%	65.22%
Beef	56.67%	66.67%	50.00%
CBF	94.22%	94.22%	97.11%
Coffee	82.14%	82.14%	82.14%
ECG200	100.00%	100.00%	100.00%
FaceAll	65.56%	80.06%	77.99%
FaceFour	70.45%	71.59%	92.05%
Fish	74.29%	90.29%	82.86%
GunPoint	95.33%	95.33%	96.67%
Lightning2	72.13%	62.30%	72.13%
Lightning7	63.01%	83.56%	75.34%
OliveOil	66.67%	66.67%	76.67%
OSULeaf	56.20%	76.45%	71.90%
SwedishLeaf	76.80%	91.04%	84.80%
SyntheticControl	92.00%	92.00%	97.67%
Trace	100.00%	100.00%	100.00%
TwoPatterns	99.85%	99.85%	99.85%
Wafer	99.98%	99.98%	100.00%
Yoga	85.30%	90.97%	89.80%
Ford			
FordA	93.11%	97.05%	96.74%
FordB	67.04%	75.56%	74.81%
Weighted average accuracy			
All sets	**89.16%**	**93.32%**	**92.75%**
Smaller sets	**89.90%**	**89.74%**	**93.63%**
Larger sets	**89.11%**	**93.56%**	**92.69%**

Name	Basic ShiftTree			ShiftForest (XV+boost)		
	Accuracy	Rank	Points	Accuracy	Rank	Points
TSC01	87.12%	1	10	92.79%	1	10
TSC02	92.91%	2	9	94.75%	2	9
TSC03	50.43%	1	10	50.74%	1	10
TSC04	98.16%	1	10	98.16%	1	10
TSC05	89.20%	4	7	88.70%	4	7
TSC06	34.74%	12	0	45.13%	11	0
TSC07	80.40%	9	2	79.60%	10	1
TSC08	63.82%	13	0	78.03%	12	0
TSC09	66.39%	12	0	76.21%	12	0
TSC10	80.27%	9	2	80.69%	9	2
TSC11	73.40%	11	0	74.80%	11	0
TSC12	94.39%	1	10	97.58%	1	10
TSC13	74.84%	3	8	94.12%	1	10
TSC14	67.41%	13	0	73.88%	13	0
TSC15	62.92%	1	10	72.21%	1	10
TSC16	85.25%	6	5	81.18%	6	5
TSC17	90.38%	8	3	90.38%	8	3
TSC18	39.82%	13	0	49.82%	7	4
TSC19	65.12%	13	0	90.93%	9	2
TSC20	41.22%	11	0	64.73%	11	0
Total	**78.24%**	**8**	**86**	**84.13%**	**6**	**93**

Fig. 4. LEFT: Results of the basic ShiftTree, boosting used 100 iterations and XV used 20 iterations and 30% of the training set as the validation set. A dataset is considered "smaller" if its training set contains less than 70 series.
RIGHT: Results of the basic ShiftTree and the ShiftForest on the blind test.

forest building technique. But if we look closer, the XV greatly outperforms boosting on some datasets. The common property of these datasets is that their training set is small. If we examine the average accuracy on datasets having less than 70 samples for training (smaller sets), even the basic method outperforms

boosting by a bit. The reason for this is that even the pruned ShiftTree model fits perfectly to the small set of training data therefore the boosting stops after one iteration. Thus we basically get the original ShiftTree algorithm back. XV uses different training sets that insures to build different models. And by averaging those models, improved accuracy can be achieved even on smaller datasets. But XV is much more simpler than boosting so if enough training data is available, boosting will surpasses XV. We found that "enough" means 70 training samples by the UCR database. By using the appropriate method for all datasets, an average accuracy of 93.57% can be achieved.

7.4 Results of the Blind Tests

As we mentioned above, we used the datasets of the SIGKDD2007 Time Series Challenge to evaluate our algorithm in a blind test. We did one test for the basic algorithm and one for the best forest method. The parameterization of the basic method is the same as in the previous subsections. The parameterization of the forest method is the same as the best parameterization for the UCR datasets: on datasets having a training set of less than 70 examples, we used XV ($M = 20$, $S = 30\%$), on the rest we used boosting with 100 as the number of iterations. We calculated the points for ShiftTree as it had taken part in the competition: 10 point for a 1^{st} place, 9 for a 2^{nd} and so on. We also modified the points of the other participants, if the ShiftTree surpassed the accuracy of their algorithms. The two methods took part in two separate tests (i.e. they were not competing each other). Figure 4 shows the results.

Out of 12+1 participants basic ShiftTree gained a total rank of 8. We consider this to be a great success as the operators/parameters were not optimized and we assume that most of the participants used blended algorithms. Interestingly, ShiftTree ranked 1^{st} place 5 times out of 20 which is the same amount as the overall winner has. We think that the reason for this is that ShiftTree is accurate on datasets with greater average training examples per class values. The ShiftTree forest improves the overall results: it gains one more 1^{st} place as the overall winner loses one and moves forward to the 6^{th} place of the challenge.

8 Conclusion

We proposed a new model-based time series classifier called ShiftTree, which is an important advance in this research field because of its unique benefits: thanks to its model-base approach, the decision tree based model can be interpretable, this property is infrequent in this data mining domain, where the instance-based algorithms are overrepresented.

The key aspect of time series classification is the handling of correspondences of time dimension. Instead of an elastic time approach, we propose a novel attribute indexing technique: a cursor is assigned to each instance of the time series dataset. By determining the cursor operators, our algorithm can classify with high accuracy. The supervised learning method of the ShiftTree is an extension

of the decision tree building methods, where the cursor operator and attribute calculator modes are designated beyond the splitting value.

The numerical results of 22 datasets of a benchmark collection show that our algorithm has similar accuracy to other techniques, moreover its accuracy exceeds the best instance-based algorithms on some datasets. As it is typical of model-based algorithms, the ShiftTree algorithm can work more efficiently than the instance-based methods when the size of training dataset is larger. The proposed forest extension of ShiftTree, the cross-validation based and boosting techniques are to improve the accuracy level of ShiftTree. The efficiency of the algorithm can be further enhanced by defining domain specific cursor operators and attribute calculators, where the specific characteristics of the dataset are also used.

Although the efficiency of our algorithm is significant in some cases, its importance lies in the fact, that by novel cursor-based attribute indexing, it can solve some classification problems which can not be answered by the other model-based or instance-based methods. We think that this attribute indexing technique is promising and it can be the base of a novel algorithm family in the future in the field of time series and spatial data mining.

References

1. Abou-Nasr, M., Feldkamp, L.: Ford Classification Challange (2008), http://home.comcast.net/~nn_classification/
2. Acir, N.: Classification of ecg beats by using a fast least square support vector machines with a dynamic programming feature selection algorithm. Neural Computing and Applications 14(4), 299–309 (2005)
3. Azoff, E.M.: Neural Network Time Series: Forecasting of Financial Markets. Wiley, Chichester (1994)
4. Pong Chan, K., Chee Fu, A.W.: Efficient time series matching by wavelets. In: ICDE (1999)
5. Chen, L., Ng, R.: On the marriage of lp-norms and edit distance. In: VLDB (2004)
6. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: SIGMOD Conference (2005)
7. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: Experimental comparison of representations and distance measures. In: VLDB (2008)
8. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: SIGMOD Conference Proceedings (1994)
9. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences (55), 119–139 (1997)
10. Jager, H.: The echo state approach to analysing and training recurrent neural networks. GMD Report 148 8, 1–42 (August 2001)
11. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. 7(3) (2005)
12. Keogh, E., Shelton, C., Moerchen, F.: Workshop and Challenge on Time Series Classification at SIGKDD 2007 (2007), http://www.cs.ucr.edu/~eamonn/SIGKDD2007TimeSeries.html

13. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering Homepage (2006), http://www.cs.ucr.edu/~eamonn/time_series_data/
14. Lin, J., Keogh, E., Lonardi, S., Patel, P.: Finding motifs in time series. In: 2nd Workshop on Temporal Data Mining (KDD 2002), pp. 53–68 (2002)
15. Prekopcsak, Z.: Accelerometer based real-time gesture recognition. In: POSTER 2008: Proceedings of the 12th International Student Conference on Electrical Engineering (May 2008)
16. Quinlan, J.R.: Induction of decision trees. In: Readings in Machine Learning. Morgan Kaufmann, San Francisco (1990)
17. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. Data Engineering (August 2002)
18. Vlachos, M., Gunopulos, D., Kollios, G.: Discovering similar multidimensional trajectories. In: ICDE (2002)
19. Zhu, J., Rosset, S., Zou, H., Hastie, T.: Multi-class adaboost. Statistics and Its Interface 2, 349–360 (2009)

Image Classification for Age-related Macular Degeneration Screening Using Hierarchical Image Decompositions and Graph Mining

Mohd Hanafi Ahmad Hijazi[1,3,⋆], Chuntao Jiang[1], Frans Coenen[1], and Yalin Zheng[2]

[1] Department of Computer Science, University of Liverpool, Liverpool L69 3BX, UK
[2] Department of Eye and Vision Science, Institute of Ageing and Chronic Disease, University of Liverpool, UCD Building, Liverpool L69 3GA, UK
[3] School of Engineering and Information Technology, Universiti Malaysia Sabah, Locked Bag 2073, 88999 Kota Kinabalu, Sabah, Malaysia
{m.ahmad-hijazi,c.jiang,coenen,yalin.zheng}@liverpool.ac.uk

Abstract. Age-related Macular Degeneration (AMD) is the most common cause of adult blindness in the developed world. This paper describes a new image mining technique to perform automated detection of AMD from colour fundus photographs. The technique comprises a novel hierarchical image decomposition mechanism founded on a circular and angular partitioning. The resulting decomposition is then stored in a tree structure to which a weighted frequent sub-tree mining algorithm is applied. The identified sub-graphs are then incorporated into a feature vector representation (one vector per image) to which classification techniques can be applied. The results show that the proposed approach performs both efficiently and accurately.

Keywords: Hierarchical image decomposition, weighted graph mining, image partitioning, image classification.

1 Introduction

Vision loss and blindness may be caused by various factors; Age-related Macular Degeneration (AMD) [17] is the leading cause of adult blindness in the developed world [21]. AMD is currently incurable and causes total blindness. There are new treatments that can stem the onset of AMD if detected at a sufficiently early stage. Drusen, sub-retinal deposits formed by retinal waste, are the first clinical indicator of AMD. The presence of drusen can be detected by inspection of retina images. Substantial work has been directed at applying image processing and content-based image retrieval techniques to support the diagnosis of AMD; however current performance of these techniques is still not sufficient for wide-scale clinical application, largely because of the limitations of the segmentation techniques adopted.

⋆ Corresponding author.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 65–80, 2011.

This paper describes an image mining approach to AMD screening where the objective is to classify images as being either AMD or non-AMD. A first attempt at data mining supported AMD screening, that employed a histogram based representation, is described in [14,13]. The objective was to avoid the segmentation difficulties encountered by previous techniques. The technique produced reasonable results. The technique described in this paper is founded on a novel interleaved angular and circular hierarchical decomposition of the image space, the aim being to isolate instances of drusen. The decomposition is stored in a tree data structure to which a weighted frequent sub-tree mining algorithm is applied. The identified frequent sub-trees are then used to define a feature space which can be used to encode an appropriately labelled training set into a set of feature vectors (one per image) to which established classification techniques can be applied. The proposed technique has been evaluated using a sample set of coloured retinal fundus images featuring both AMD images and a control group.

The main contributions of this paper are:

- The proposed circular and angular based hierarchical decomposition.
- The mechanism for generating feature vectors using weighted frequent sub-tree mining.
- The application of the above techniques to AMD screening.

The rest of this paper is organised as follows: some previous works with respect to image decomposition approaches and weighted frequent sub-graph mining is described in Section 2. Section 3 provides a description of the AMD application domain, followed by details of the proposed image classification approach in Section 4. The performance of the proposed approach is extensively evaluated in Section 5 and some conclusions are presented in Section 6.

2 Previous Work

Hierarchical data structures have been widely applied in various domains, such as image segmentation [25], image coding [12] and image classification [9]. The main advantage of this type of data structure is that it provides an effective representation of the problem domain that can be readily processed [23]. The most common hierarchical decomposition technique is founded on quadtrees, where the search space is repeatedly quartered until uniform "tiles" are arrived at or a maximum decomposition is reached. In the work described in this paper a new image decomposition technique more suited to retinal images, that uses an alternating angular and circular partitioning, is presented.

Graph mining techniques can be categorised as being either *transaction* based or *single* graph based [18]. Transaction graph mining aims to discover frequently occurring sub-graphs in a given graph data set. Weighted frequent sub-graph mining is founded on the idea that in some cases certain vertices and/or edges in the input graph set can be deemed to be more significant than others. The weighted frequent sub-graph mining has demonstrated its advantages over frequent sub-graph mining in a number of studies [9,18], the main advantage is

that the former spends significantly less run time identifying far fewer patterns (i.e. frequent sub-graphs) than the latter. A variation of a weighted frequent sub-graph mining algorithm [18], founded on the well-known gSpan algorithm [27], is used with respect to the work described in this paper and is described in detail in Subsection 4.2.

Image processing approaches have been widely used in the detection of drusen for AMD diagnosis. The earliest work [24] used a morphological mechanism to localise drusen. Other image processing techniques that have been applied include: (i) histogram-based adaptive local thresholding [22] , (ii) region growing [19,20]; (iii) wavelet based feature identification coupled with multilevel classification [3]; (iv) anomaly detection based approaches, that employ Support Vector Data Description (SVDD), to segment anomalous pixels [11]; and (v) signal based approaches, namely amplitude-modulation frequency-modulation (AM-FM), to generate multi-scale features for drusen classification [1,2]. Content-Based Image Retrieval (CBIR) techniques have also been applied. For example, Chaum et al. [6] have applied CBIR to get a probability of the presence of a particular pathology (a confidence threshold is then applied on the generated probabilities to predict the retinal images class).

Most of the existing work on AMD diagnosis is founded on drusen detection and segmentation. The authors are aware of only three reports of extending drusen detection to AMD screening [2,3,6]. One issue is the difficulties and challenges in not only identifying the drusen, but also other retinal structures, in particular the optic disc and the macula. The challenge is exacerbated by the natural variation of the appearance of the retina, the image quality and patient factors (e.g. compliance during image acquisition and media clarity of the eye) [6,15]. Alternative techniques, not founded on segmentation, therefore seem desirable. An early attempt in this direction has been proposed in [13] whereby a histogram based representation was used to which Case-Based Reasoning (CBR) was applied to facilitate the classification of "new" cases. Another approach that was built based on work in [13] that uses two "case bases" has been proposed in [15]. Good results were produced, however observations indicated that relying on the retinal image colour distribution alone was not sufficient. Thus, in [14] a spatial histogram technique, that include colour and spatial information, was proposed. The technique gave the best results so far with respect to the test image dataset. The work described in this paper is directed at improving on these results.

3 Age-related Macular Degeneration

Some exemplar retinal images are presented in Figure 1. The central region of the human retina is called the *Macula*, which is centered at the *fovea*; this is the place where acute and central vision is made possible. Figure 1(a) shows the fovea (indicated by an arrow) and macula (circled). Damage to the macula causes distortion and loss of the central vision required for (say) reading and writing. There are various reasons why this might happen, one of which is Age-related Macular Degeneration (AMD) where the delicate cells of the macula

becomes damaged and stops functioning at the later stages of life [17]. AMD can be categorised into *early*, *intermediate* and *advanced*. Advanced AMD, which can be further divided into *non-neovascular* and *neovascular*, is when severe vision loss or even total blindness occurs [17]. Although AMD is incurable, the early detection of AMD is desirable as there are new emerging treatments that can be given to patients with AMD to slow down or even halt the progress of the condition.

As noted above, early stage AMD can be diagnosed by the existence of *drusen* [8,17], yellowish-white deposits with sizes ranging from less than 63 μm (*small* drusen) to greater than 124 μm (*large* drusen) in diameter [17]. These can be detected through manual inspection, by trained clinicians, of the retinal images collected within screening programmes. This task is however labour intensive given the increasing incidence of AMD. Thus, the automation or semi automation of the process is deemed desirable. Figure 1(b) shows a case of intermediate AMD where the presence of drusen (surrounded by circle) is widespread but the fovea is still visible (dark area pointed by arrow). Figure 1(c) gives an example of advanced neovascular AMD where the fovea is totally obscured resulting in total loss of central vision. Drusen can also be categorised into *hard* or *soft* drusen. Hard drusen is more easy to identify because it has well defined borders, while soft drusen has indistinct edges that blend into the retinal background.

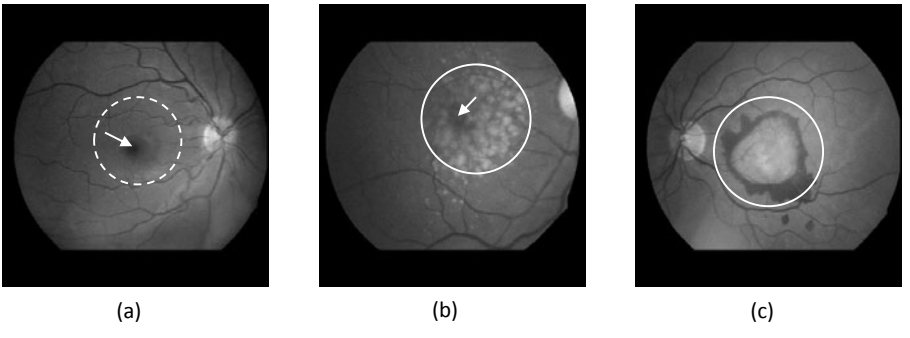


Fig. 1. Grayscaled retinal fundus images: (a) normal, (b) intermediate, and (c) advanced neovascular

4 AMD Classifier Generation

An overview of the proposed hierarchical decomposition based approach to the generation of AMD classifiers is presented in Figure 2. The approach commences with retinal image cleaning. The quality of the retinal fundus images is often affected by various factors that hinder image classification, such as colour variation and nonuniform illumination. Some image cleaning is therefore required. In the context of the work described in this paper, an approach used in [14] is reused to enhanced the images and to identify blood vessel pixels (which we wish to remove from the image data so as to "clean" the data). The process then proceeds

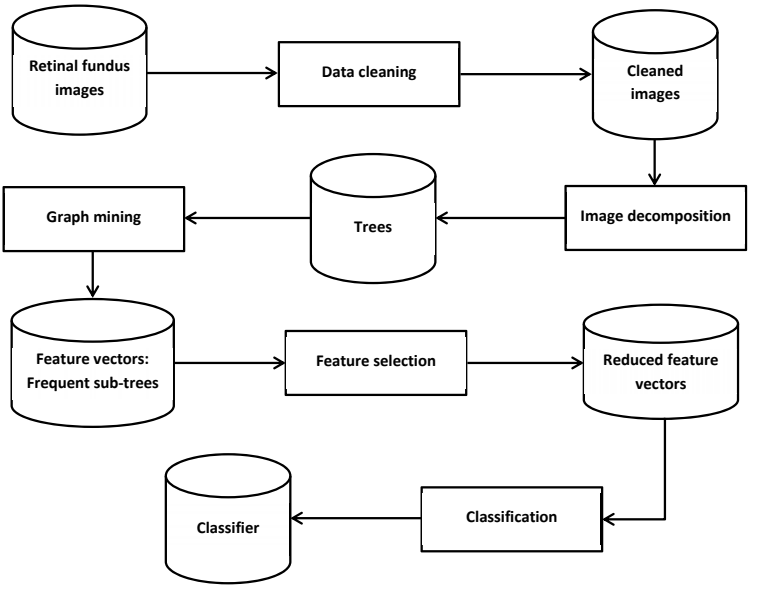


Fig. 2. Block diagram of the proposed classifier generation approach

with the decomposition of the image, this is described in detail in Subsection 4.1 below; the result is a collection of tree represented images (one per image). Next the weighted frequent sub-graph (sub-tree) mining approach is applied to the data (the algorithm is detailed in Subsection 4.2). The identified frequent sub-trees then define the elements of a feature space that is used to encode the individual input images in the form of feature vectors itemising the frequent sub-graphs that occur in each image. The feature selection process is described in Subsection 4.3. Once the feature vector representation has been generated we can apply established classification techniques (see Subsection 4.4).

4.1 Image Decomposition

The proposed image decomposition method is described in this sub-section. As noted above hierarchical image decomposition is a well established technique [12,23,25]. The distinguishing and novel feature of the proposed approach is that the partitioning is conducted in an interleaving *angular* and *circular* manner. During angular partitioning the decomposition is defined by two radii describing a minor arc on the circumference of the image "disc". Circular decomposition is defined by a pair of arcs radiating out from the center of the retina disc. Individual regions identified during the decomposition are thus delimited by a pair of radii and a pair of arcs. Figure 3(a) shows an example of a partitioning that might be applied to an image; Figure 3(b) presents the associated *Tree* storage structure. Note that a numbering convention is used to label individual regions described by nodes in the tree structure.

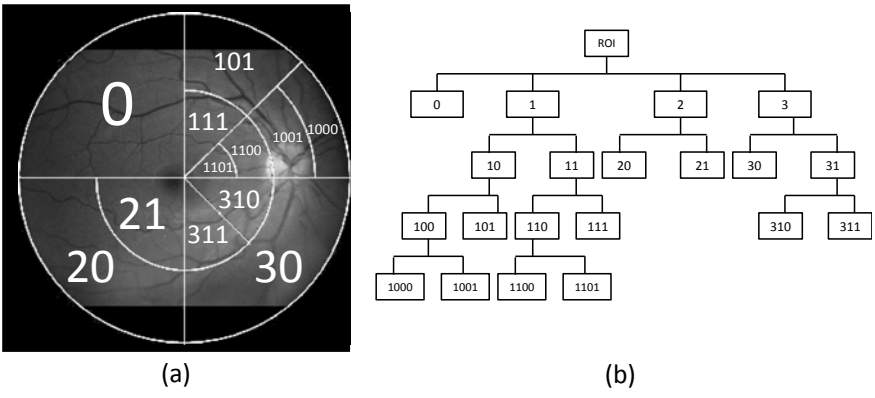


Fig. 3. An example of: (a) circular and angular image decomposition, and (b) the associate tree data structure

Algorithm 1 shows how the interleaved circular and angular partitioning is performed. Given a coloured retinal fundus image, I with a size of X pixels. The RGB (red, green and blue) colour model is used to extract the pixels intensity values, which means each pixel will have three intensity values (red, green, blue) associated with it, hence initially three trees are generated which are then merged. The *GetCentroids* method in line 4 uses a retinal image mask, M, to identify the centroid of the retina disc. The *GetImageBackground* method in line 5 generates a binary format background image, *imbg*, to be used to distinguish the background (areas outside of the field of view of the fundus) pixels and the blood vessel pixels, V, from the retinal pixels. *imbg* is defined as:

$$imbg = M \cap RV \tag{1}$$

$$M(x) = \begin{cases} 1, & \text{if } x \text{ is a retina pixel} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$RV(x) = \begin{cases} 0, & \text{if } x \text{ is a blood vessels pixel,} \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

where $x \in X$, and M and RV are both of size of X pixels. The image ROI was then identified using the *GetROI* method.

As noted above the proposed hierarchical image partitioning commences with *AngularPartitioning* (line 13 of Algorithm 1). On the next iteration *CircularPartitioning* will be applied. Both *AngularPartitioning* and *CircularPartitioning* will then be called alternately until the D_{max} tree level is reached or only regions of uniform intensity are left. Throughout the process the tree data structure is continuously appended to. The algorithm ends with the merging of the three trees in T using the *MergeTrees* method to form a single tree. The merging is done by calculating the Average Intensity Values (AIV) for the nodes in T, defined as:

$$AIV_y = \frac{1}{n}\sum_{k=1}^{3}(T_{k_y}) \tag{4}$$

$$T_{k_y} = \frac{1}{z}\sum_{i=1}^{z}(k_{y_i}) \tag{5}$$

where T comprises the red, green and blue colour trees, y is a unique node identifier in T_{final}, z is number of pixels in node y, and n is number of occurrences of node y (whenever T_{k_y} is not null) in the set of trees T.

Algorithm 1: ImageDecomposition

Data: Coloured retinal fundus image I, retinal image mask M, retinal blood vessels binary image RV and D_{max}
Result: Image decomposition tree T_{final}

```
1  c_count ← 1;
2  a_count ← 1;
3  T ← {null, null, null};
4  centroid ← GetCentroid(M);
5  imbg ← GetImageBackground(M, RV);
6  roi ← GetROI(imbg, centroid);
7  for k ← 0 to 2 do                    // Generate trees for each colour channel
8      for i ← 1 to maxDepth do         // Generate trees for each tree level
9          if mod(i/2) = 0 then
10             t ← CircularPartitioning(roi, imbg, c_count, centroid);
11             c_count ← c_count + 1;
12         else
13             t ← AngularPartitioning(roi, imbg, a_count, centroid);
14             a_count ← a_count + 1;
15         end
16         tree ← UpdateTree(tree, t);
17     end
18     T_k ← tree;
19 end
20 T_final ← MergeTrees(T_0, T_1, T_2);
```

Algorithm 2 describes the *CircularPartitioning* method. The algorithm returns a set of new nodes B to be added to the tree structure. The input is the ROI image *roi*, the mask *imbg*, the level count for the circular partitioning (*c_count*) and the centroid of the ROI (retina disk). We first identify how many circles, m are required for the current iteration (line 1). Then we calculate a set of new radii to be included in the partitioning; $R = \{\rho_0, \rho_1, \ldots, \rho_m\}$ describes a sequence of concentric circles. We then (line 3) generate the necessary additional nodes for the tree. Each level *c_count* region is considered in turn and, where appropriate, *c_count* + 1 regions are constructed.

Algorithm 2: CircularPartitioning

Data: Retinal image *roi*, image background *imbg*, *c_count* and *centroid*
Result: An array of circular partitioned image regions B

1 $m \leftarrow 2^{c\_count}$; // To calculate number of circles
2 $R \leftarrow GetRadius(imbg, centroid, m)$; // To calculate radii values
3 $B \leftarrow SplitImage(roi, imbg, R, centroid)$;
4 **return** B

The *AngularPartitioning* method, as described in Algorithm 3, begins by identifying the number of radii (m) that are required (line 1). The radii define the angular partitions which are defined in terms of a set of arcs $A = \{\alpha_0, \alpha_1, \ldots, \alpha_m\}$. Each arc α is defined by $\theta = 2\pi/m$, used in the *GetTheta* method (line 2). As in the case of the *CircularPartitioning* algorithm, the *SplitImage* method is then called to decompose the image, as indicated, to produce an appropriate set of nodes B.

Algorithm 3: AngularPartitioning

Data: Retinal image *roi*, image background *imbg*, *a_count* and *centroid*
Result: An array of angular partitioned image regions B

1 $m \leftarrow 2 \times 2^{a\_count}$; // To calculate number of angular lines
2 $\theta \leftarrow GetTheta(n)$; // To calculate the angle between angular lines
3 $B \leftarrow SplitImage(roi, imbg, n, \theta, centroid)$;
4 **return** B

As noted above, an important feature of the hierarchical image decomposition is the selection of a termination criterion, ω, which defines the homogeneity of a particular region in an image and is used to determine if further region splitting is required. A common definition for ω is in terms of the distance between the highest and lowest intensity values of the pixels in a region i. If ω is less than a predefined homogeneity threshold, τ, no further decomposition of the region i will be undertaken.

In this paper, a similar termination criterion to that described in [12] is adopted. The ω value is defined according to how well a parent region represents its two child regions' intensity values. If the value (derived from the average intensity values of all pixels in a particular region) of a parent region is similar ($< \tau$) to that of its two child regions, the parent region is deemed to be homogeneous and left unsplit. Otherwise, the parent region will be further partitioned. Thus ω can be formalised as:

$$\omega = \frac{1}{s} \sum_{i=1}^{s} \sqrt{(\mu_p - \mu_i)^2} \tag{6}$$

where s is the number of sub-regions, μ_p is the average intensity value for the parent region and μ_i is the average intensity value for sub-region i. Each

identified sub-region is represented as a "node" in a tree data structure where the relationship between sub-regions and their parent form the tree "edges".

4.2 Weighted Frequent Sub-tree (wFST) Mining

After the image decomposition step introduced in the above subsection, images were modelled as a collection of trees. According to [18], each tree is defined as follows: $T = \{V, E, L_V, L_E, \phi\}$, where L_V and L_E are labels for nodes and edges in T respectively, and ϕ defines a label mapping function.

In [18] it was suggested that for many applications, such as image mining, some tree nodes have more significance associated with them than other nodes. In the case of the hierarchical decomposition described in this paper, nodes that feature a significant difference in colour intensity when compared to their parent node are deemed to be more significant (than the parent). The underpinning philosophy here is that normal retinal background pixels have a similar colour intensity, while a significant difference in intensity is likely to indicate the presence of drusen. A weighting scheme was therefore applied to the tree representation so as to enhance the quality of the information contained within it. Thus, in the tree representation, the strength of each node $v \in V$ was weighted by the average colour intensity value of the region represented by that node v, and the strength of each edge $e \in E$, e_w, is weighted by:

$$e_w = \sqrt{(I_{par} - I_v)^2} \tag{7}$$

where I_v is the average colour intensity value for node v and I_{par} is the average colour intensity value for v's parent.

By adding node and edge weights into the tree representation, the weighted tree representation was able to capture more image information than the unweighted one. A weighted Frequent Sub-Tree (wFST) mining algorithm, an extension of the well-known gSpan algorithm [27], was then applied to the tree data so as to identify frequently occurring trees within the dataset. The wFST algorithm operated in a similar manner to that described in [18], but utilised both node and edge weightings. In the context of wFST mining, a sub-graph pattern g is considered to be "interesting", if it satisfies the following two conditions:

$$(\mathbf{C1}) N_{wr} \times sup(g) \geq \sigma, \qquad (\mathbf{C2}) E_{wr} \geq \lambda \tag{8}$$

Where: N_{wr} denotes the node weighting, $sup(g)$ denotes the support (i.e. frequency) of g, and σ denotes a *minimum support* threshold, E_{wr} denotes the edge weighting, and λ denotes a *minimum weight* threshold. Both the N_{wr} and E_{wr} are computed using a similar scheme to that described in [18].

The number of patterns discovered by the wFST mining algorithm is thus determined by both the σ and λ values. According to initial experiments conducted by the authors, relatively low σ and λ values are required, in order to extract a sufficient number of image features (frequent sub-trees). However, setting low threshold values results in a substantial number of patterns, of which many are

redundant in terms of the desired classification. Therefore, feature selection was applied to the discovered patterns, this is discussed in the following subsection.

4.3 Feature Selection

For the AMD application the number of identified wFSTs was substantial, using low threshold values tens of thousands of wFSTs were identified. To reduce the number of wFSTs to a manageable number a feature selection strategy was applied so as to identify those wFSTs that displayed a strong discriminatory power, which would consequently be able to produce good classification results. A feature ranking mechansim was therefore used, with respect to the AMD application, that used linear Support Vector Machine (SVM) weights to rank features as proposed in [5]. The main advantage of this approach is its implementation simplicity and effectiveness in determining relevant features. The identified wFSTs were ranked by first calculating their weights using the L2-regularized L2-loss SVM model, and then sorting them in descending order according to their absolute value [10,16]. The selection of only the top K wFSTs for classification then concluded the feature selection process.

4.4 Classification Technique

The final stage of the proposed retinal image classification process was the classification stage. The identified top K wFSTs were used to define a feature space. Each image was then defined, in terms of this feature space, using a feature vector representation. Any appropriate classification technique could then be applied. In the reported experiments (Section 5), two different classifier generation techniques were used, Naïve Bayes [26] and Support Vector Machine (SVM) [7]. Naïve Bayes was selected because: (i) it works very well when tested on data with independent attributes [26], and (ii) it does not require user defined parameters. SVM on the other hand was chosen because it is frequently acknowledged to be one of the most effective classification method in machine learning. In this paper, the Naïve Bayes classifier available in Weka [26] was used. The second classifier was built using LibSVM [4] with a radial basis function kernel.

5 Evaluation

The proposed AMD screening approach was applied to two retinal fundus images datasets, ARIA[1] and STARE[2]. The ARIA dataset comprises 161 images, of which 101 were AMD and 60 were normal images. The STARE dataset comprised 97 images, of which 59 were AMD and 38 were normal images. The datasets were merged to create an image set comprising 258 images, of which 160 featured AMD and 98 were normal. Sensitivity, specificity and accuracy were used to measure the classification performance.

[1] http://www.eyecharity.com/aria_online/
[2] http://www.ces.clemson.edu/~ahoover/stare

In the experiments three values for D_{max} were used: 5, 6 and 7. The threshold for node splitting, τ, was set to 2.5%. To train the LibLINEAR (for feature selection task) and LibSVM classifiers, the default values for the user defined parameters used in Weka were applied, except for the soft margin parameter C and gamma parameter γ which were determined using the parameter selection tool provided with LibLINEAR and LibSVM [4]. The aim of the experiments was to evaluate: (i) the effect of the value of D_{max} on the classification results, (ii) how feature selection improved the classification performance, and (iii) how well the proposed approach's performance compared with other AMD classification techniques. Most of the reported experiments were conducted using Ten-fold Cross Validation (TCV).

5.1 Performances Using Different Levels of Decomposition

Tables 1 and 2 shows the performances of the proposed approach when using the three different levels of decomposition (values for D_{max}) and using Naïve Bayes and LibSVM respectively. Feature selection was not applied in these experiments. Minimum support, σ was used to prune the candidate sub-trees, while the minimum weight, λ was used to further reduce the number of identified frequent sub-trees according to their edge weights. F denotes the size of the feature space in terms of the number of identified frequent sub-trees, while *Sens*, *Spec* and *Acc* refers to sensitivity, specificity and accuracy. Each σ value was tested against a range of λ values (20, 40, 60 and 80), however in the table (because of space limitations) only the best performing λ value associated with each σ value is recorded.

Table 1. TCV Classification results obtained using different levels of decomposition and Naïve Bayes

σ	**5**					**6**					**7**				
(%)	λ	F	*Sens*	*Spec*	*Acc*	λ	F	*Sens*	*Spec*	*Acc*	λ	F	*Sens*	*Spec*	*Acc*
10	40	764	65	60	**63**	20	7125	64	53	60	80	3433	66	42	57
20	60	291	68	51	62	80	498	70	42	59	80	3433	66	42	57
30	60	291	68	51	62	20	1746	66	49	60	80	3433	66	42	57
40	20	248	66	48	59	80	498	70	42	59	80	3433	66	42	57
50	20	181	69	45	60	80	498	70	42	59	80	3433	66	42	57
60	20	130	67	46	59	20	559	69	45	60	20	3893	66	42	57
70	20	99	69	33	55	20	404	68	34	55	20	2623	69	41	58
80	20	71	80	13	55	20	283	72	27	55	20	1706	70	36	57
90	20	55	86	16	60	20	180	79	22	58	20	955	72	28	55

Inspect of Tables 1 and 2 indicates that the best accuracy was achieved using $D_{max} = 5$ (63% and 70%). The best sensitivity and specificity were 86% and 60% (Naïve Bayes) and 100% and 43% (LibSVM) respectively. The best sensitivity and specificity for both classifiers occurred using different σ and λ values (but again with $D_{max} = 5$). Experiments were also conducted using an unweighted FST mining algorithm (gSpan), however these indicated that the memory requirements and runtime deemed to be unacceptable.

Table 2. TCV Classification results obtained using different levels of decomposition and LibSVM

σ	**5**					**6**					**7**				
(%)	λ	F	$Sens$	$Spec$	Acc	λ	F	$Sens$	$Spec$	Acc	λ	F	$Sens$	$Spec$	Acc
10	40	764	86	43	**70**	20	7125	87	39	69	60	11461	96	15	66
20	20	594	89	37	69	20	3103	91	30	68	60	11461	96	15	66
30	20	365	95	16	65	60	1358	89	33	68	60	11461	96	15	66
40	80	118	100	0	62	20	1135	92	30	68	20	9043	97	8	63
50	80	118	100	0	62	20	779	94	11	62	80	3433	96	8	63
60	20	130	83	30	63	20	559	99	1	62	20	3893	96	11	64
70	20	99	99	4	63	20	404	98	0	60	20	2623	99	4	63
80	20	71	100	0	62	20	283	100	1	62	20	1706	97	8	63
90	20	55	99	3	63	20	180	100	0	62	20	955	96	10	63

5.2 Performances of AMD Classification According to the Size of the Identified Feature Space

Tables 3 and 4 shows the performances of the proposed approach with respect to different values of K, using Naïve Bayes and LibSVM respectively. Recall that the size of the feature space was determined by selecting only the top K features defined as a percentage (P) of $|F|$ where F is the set of features. Experiments using five different P values were conducted: 0.05, 0.1, 0.2, 0.4 and 0.6. However, only the results using a $D_{max} = 7$ and P values of 0.05, 0.1 and 0.4 are presented in the tables because these produced the best classification performances with respect to both classifiers. Inspection of the tables indicates how the performance changes as the size of the feature space is reduced.

Table 3. TCV Classification results using feature selection, a decomposition level (D_{max}) of 7, and Naïve Bayes

σ	$P_{0.05}$					$P_{0.1}$					$P_{0.4}$				
(%)	λ	K	$Sens$	$Spec$	Acc	λ	K	$Sens$	$Spec$	Acc	λ	K	$Sens$	$Spec$	Acc
10	20	3671	94	96	**95**	20	7342	92	92	92	40	16278	73	67	71
20	20	1407	91	93	91	20	2814	88	82	85	20	11257	71	65	68
30	20	748	88	82	85	20	1496	86	78	83	20	5983	71	61	67
40	20	452	85	80	83	20	904	84	72	79	20	3618	71	56	65
50	20	291	85	72	80	80	343	86	64	78	20	2330	73	52	65
60	20	195	84	68	78	20	389	86	67	78	20	1558	76	51	66
70	20	131	83	63	75	20	262	83	56	72	20	1050	74	47	64
80	20	85	82	45	68	20	171	81	50	69	20	683	74	45	63
90	20	48	83	37	65	20	96	84	41	67	20	382	78	37	62

Tables 3 and 4 demonstrate that the best results were obtained using lower numbers of features, where $P = 0.05$ and $K = 3671$. The best accuracy for Naïve Bayes was 95% while LibSVM recorded a full 100% accuracy. The highest sensitivity and specificity was 100% (LibSVM). High sensitivity and specificity were also achieved using the Naïve Bayes classifier with a corresponding accuracy of 94% and 96% respectively. All of the best results were generated using $\sigma = 10\%$

Table 4. TCV Classification results using feature selection, a decomposition level (D_{max}) of 7, and LibSVM

σ	$P_{0.05}$					$P_{0.1}$					$P_{0.4}$				
(%)	λ	K	*Sens*	*Spec*	*Acc*	λ	K	*Sens*	*Spec*	*Acc*	λ	K	*Sens*	*Spec*	*Acc*
10	20	3671	100	100	**100**	20	7342	100	100	**100**	40	16278	99	81	92
20	80	172	100	0	62	80	343	100	0	62	60	4585	99	8	65
30	20	748	99	80	92	20	1496	99	94	97	20	5983	98	70	87
40	80	172	100	0	62	80	172	100	0	62	80	1374	99	5	64
50	20	291	97	54	81	20	583	96	84	91	20	2330	95	56	80
60	80	172	100	0	62	80	172	100	0	62	80	1374	99	5	64
70	20	131	100	0	62	20	262	100	0	62	20	1050	100	2	63
80	20	85	100	0	62	20	171	100	1	62	20	683	99	6	64
90	20	48	100	0	62	20	96	100	1	62	20	382	98	10	65

and $\lambda = 20\%$ were applied. The accuracy increased as the σ value decreased for all K values. The results produced show that the larger the feature space the better the classification performance. It should be noted that the results reported in Subsection 5.1, where feature selection was not applied are not as good as those reported here. Feature selection clearly improves the classification performance.

5.3 Performance Comparison of AMD Classification Using Various Classification Techniques

Table 5 compares the classification results obtained using the spatial histogram based approach [14] referred to earlier in Section 2 and the AMD screening approach proposed in this paper using the LibSVM classifier. Both approaches were applied to the ARIA dataset. The parameters were set to: $\sigma = 30\%$, $\lambda = 20\%$ and $D_{max} = 7$. These values were chosen as they produced the best classification results through series of experiments conducted. The results clearly indicate the superiority of the proposed approach.

Table 5. Comparison of proposed AMD screening approach with alternative histogram based approach

Approach	**Features**	**Sensitivity**	**Specificity**	**Accuracy**
Histogram based [14]	640	86	56	74
Proposed approach	1354	100	100	100

Table 6 compares the performance of the proposed approach with other reported approaches: Barriga et al. [2], Chaum et al. [6] and Brandon and Hoover [3]. For comparison we also used the leave-one-out testing method used by them. The reported results in [2] were generated using the drusen classification technique described in [1]. For this experiment, the proposed approach utilised the

Table 6. Comparison of proposed AMD screening approach with alternative approaches

Approach	Dataset size	Sensitivity	Specificity	Accuracy
Barriga et al. [2]	100	75	50	-
Brandon and Hoover [3]	97	-	-	87
Chaum et al. [6]	395	-	-	88
Proposed approach	258	89	99	93

Naïve Bayes classifier with $\sigma = 10\%$ and $\lambda = 20\%$ ($K = 3671$). A decomposition level of $D_{max} = 7$ was used.

Table 6 includes some missing values because these were not reported in the literature and could not be derived by the authors. The result obtained by Barriga et al. [2] only reported sensitivity and specificity. On the other hand, the work of Brandon and Hoover [3] only reported accuracy[3], no sensitivity and specificity values were reported. It should also be noted that the approach of Chaum et al. [6] was actually applied in a multi-class setting, of which 12 of the AMD images were classified as "unknown" and excluded from the accuracy calculation (if included this would give an accuracy of 75%). Overall, the results demonstrate that the proposed AMD screening approach outperforms the other approaches by 14% (sensitivity), 49%(specificity) and 5% (accuracy).

6 Conclusions

An AMD screening approach founded on a hierarchical circular and angular image decomposition technique has been described. The decomposition results in a tree data structure to which a weighted tree mining technique was applied so as to identify frequent occurring sub-trees. The generated weighted frequent sub-trees were then used to recast the input data (the training set) into a feature vector representation. AMD classifiers were then built using the feature vector representation as the input data. For evaluation purpose the proposed approach was applied to retinal fundus images data from two publically available databases. A 100% accuracy was produced using the LibSVM classifier. A more straightforward and parameter-free classification technique, Naïve Bayes, was also experimented with and also generated good results (95% accuracy). Further experiments demonstrated the superiority of the proposed approach compared to some other reported techniques for AMD detection. Our current work is directed at extending the proposed approach to address multi-class problems. The authors are also interested in grading the severity of AMD, as well as using the approach for the screening of other retinal diseases, such as diabetic retinopathy.

[3] http://www.parl.clemson.edu/stare/drusen/

References

1. Barriga, E.S., Murray, V., Agurto, C., Pattichis, M.S., Russell, S., Abramoff, M.D., Davis, H., Soliz, P.: Multi-scale AM-FM for lesion phenotyping on age-related macular degeneration. In: IEEE International Symposium on Computer-Based Medical Systems, pp. 1–5 (2009)
2. Barriga, E.S., Murray, V., Agurto, C., Pattichis, M.S., Russell, S., Abramoff, M.D., Davis, H., Soliz, P.: Automatic computer-based grading for age-related maculopathy. Investigative Ophthalmology and Visual Science 51, E–Abstract 1793 (2010)
3. Brandon, L., Hoover, A.: Drusen detection in a retinal image using multi-level analysis. In: Ellis, R.E., Peters, T.M. (eds.) MICCAI 2003. LNCS, vol. 2878, pp. 618–625. Springer, Heidelberg (2003)
4. Chang, C-C., Lin, C-J.: LIBSVM: A library for support vector machines (2001) Software, http://www.csie.ntu.edu.tw/~cjlin/libsvm
5. Chang, Y.-W., Lin, C.-J.: Feature ranking using linear SVM. In: WCCI 2008, pp. 53–64 (2008)
6. Chaum, E., Karnowski, T.P., Priya Govindasamy, V.: Automated diagnosis of retinopathy by content-based image retrieval. Retina 28(10), 1463–1477 (2008)
7. Cortes, C., Vapnik, V.: Support -vector network. Machine Learning 20, 273–297 (1995)
8. de Jong, P.T.V.M.: Age-related macular degeneration. The New England Journal of Medicine 355(14), 1474–1485 (2006)
9. Elsayed, A., Coenen, F., Jiang, C., Garcia-Finana, M., Sluming, V.: Corpus callosum MR image classification. Knowledge Based Systems 23(4), 330–336 (2010)
10. Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J.: Liblinear: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
11. Freund, D.E., Bressler, N., Burlina, P.: Automated detection of drusen in the macula. In: Proceedings of the Sixth IEEE International Conference on Symposium on Biomedical Imaging: From Nano to Macro, pp. 61–64 (2009)
12. Golchin, F., Paliwal, K.K.: Quadtree-based classification in subband image coding. Digital Signal Processing 13, 656–668 (2003)
13. Hijazi, M.H.A., Coenen, F., Zheng, Y.: Image classification using histograms and time series analysis: A study of age-related macular degeneration screening in retina image data. In: Proceedings of 10th Industrial Conference on Data Mining, pp. 197–209 (2010)
14. Hijazi, M.H.A., Coenen, F., Zheng, Y.: Retinal image classification for the screening of age-related macular degeneration. In: The 30th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, pp. 325–338 (2010)
15. Hijazi, M.H.A., Coenen, F., Zheng, Y.: Retinal image classification using a histogram based approach. In: Proceedings of International Joint Conference on Neural Network 2010 (World Congress on Computational Intelligence 2010), pp. 3501–3507 (2010)
16. Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Sathiya Keerthi, S., Sundararajan, S.: A dual coordinate descent method for large-scale linear SVM. In: International Conference on Machine Learning, pp. 408–415 (2008)
17. Jager, R.D., Mieler, W.F., Mieler, J.W.: Age-related macular degeneration. The New England Journal of Medicine 358(24), 2606–2617 (2008)

18. Jiang, C., Coenen, F.: Graph-based image classification by weighting scheme. In: AI 2008, pp. 63–76 (2008)
19. Köse, C., Şevik, U., Gençalioğlu, O.: A statistical segmentation method for measuring age-related macular degeneration in retinal fundus images. Journal of Medical Systems 34(1), 1–13 (2008)
20. Köse, C., Şevik, U., Gençalioğlu, O.: Automatic segmentation of age-related macular degeneration in retinal fundus images. Computers in Biology and Medicine 38, 611–619 (2008)
21. Minassian, D., Reidy, A.: Future sight loss UK (2): An epidemiological and economic model for sight loss in the decade 2010-2020. Technical report, Royal National Institute of Blind People (2009)
22. Rapantzikos, K., Zervakis, M., Balas, K.: Detection and segmentation of drusen deposits on human retina: Potential in the diagnosis of age-related macular degeneration. Medical Image Analysis 7, 95–108 (2003)
23. Samet, H.: The quadtree and related hierarchical data structures. ACM Computing Surveys 16(2), 187–260 (1984)
24. Sbeh, Z.B., Cohen, L.D., Mimoun, G., Coscas, G.: A new approach of geodesic reconstruction for drusen segmentation in eye fundus images. IEEE Transactions on Medical Imaging 20(12), 1321–1333 (2001)
25. Spann, M., Wilson, R.: A quad-tree approach to image segmentation which combines statistical and spatial information. Pattern Recognition 18, 257–269 (1985)
26. Witten, I., Frank, E.H.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco (2005)
27. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: IEEE Conference on Data Mining, pp. 721–724 (2002)

Online Structure Learning for Markov Logic Networks

Tuyen N. Huynh and Raymond J. Mooney

Department of Computer Science, University of Texas at Austin,
1616 Guadalupe, Suite 2.408, Austin, Texas 78701, USA
{hntuyen,mooney}@cs.utexas.edu

Abstract. Most existing learning methods for Markov Logic Networks (MLNs) use batch training, which becomes computationally expensive and eventually infeasible for large datasets with thousands of training examples which may not even all fit in main memory. To address this issue, previous work has used online learning to train MLNs. However, they all assume that the model's structure (set of logical clauses) is given, and only learn the model's parameters. However, the input structure is usually incomplete, so it should also be updated. In this work, we present OSL—the first algorithm that performs both online structure and parameter learning for MLNs. Experimental results on two real-world datasets for natural-language field segmentation show that OSL outperforms systems that cannot revise structure.

1 Introduction

Statistical relational learning (SRL) concerns the induction of probabilistic knowledge that supports accurate prediction for multi-relational structured data [9]. Markov Logic Networks (MLNs) [28], sets of weighted clauses in first-order logic, are a recently developed SRL model that generalizes both full first-order logic and Markov networks which makes MLNs an expressive and powerful formalism. MLNs have also been successfully applied to a variety of real-world problems [4].

However, all existing methods for learning the structure (i.e. logical clauses) of an MLN [13,21,1,14,15] are batch algorithms that are effectively designed for training data with relatively few *mega-examples* [20]. A mega-example is a large set of connected facts, and mega-examples are disconnected and independent from each other. For instance, in WebKB [30], there are four mega-examples, each of which contains data about a particular university's computer-science department's web pages of professors, students, courses, research projects and the hyperlinks between them. However, there are many real-world problems with a different character — involving data with thousands of smaller structured examples. For example, a standard dataset for semantic role labeling consists of $90,750$ training examples where each example is a verb and all of its semantic arguments in a sentence [2]. In addition, most existing weight learning methods for MLNs employ batch training where the learner must repeatedly run inference

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 81–96, 2011.

over all training examples in each iteration, which becomes computationally expensive for datasets with thousands of training examples. To address this issue, previous work has applied online learning to set MLN weights [29,22,11]; however, to the best of our knowledge, there is no existing online *structure* learning algorithm for MLNs.

In this work, we present the first online structure learner for MLNs, called OSL, which updates both the structure and parameters. At each step, based on the model's incorrect predictions, OSL finds new clauses that fix these errors, then uses an adaptive subgradient method with l_1-regularization to update weights for both old and new clauses. Experimental results on natural language field segmentation on two real-world datasets show that OSL is able to find useful new clauses that improve the predictive accuracies of well-developed MLNs.

The remainder of the paper is organized as follows. Section 2 provides some background on MLNs and the field segmentation task. Section 3 presents our proposed algorithm. Section 4 reports the experimental evaluation on two real-world datasets. Section 5 and 6 discuss related and future work, respectively. Section 7 presents our conclusions.

2 Background

2.1 Terminology and Notation

There are four types of symbols in first-order logic: constants, variables, predicates, and functions [8]. Here, we assume that domains do not contain functions. Constants represent entities in the domain and can have types. Variables range over entities in the domain. Predicates represent properties and relations in the domain and each has a fixed number of arguments. Each argument can have a type specifying the type of constant that can fill it. We denote constants by strings starting with upper-case letters, and variables by strings starting with lower-case letters. A term is a constant or a variable. An atom is a predicate applied to terms. A ground atom is an atom all of whose arguments are constants. A positive literal is an atom, and a negative literal is a negated atom. A (possible) world is an assignment of truth values to all ground atoms in a domain. A formula consists of literals connected by logical connectives (i.e. $\vee$ and $\wedge$). A formula in clausal form, also called a clause, is a disjunction of literals.

For mathematical terms, we use lower case letters (e.g. η, λ) to denote scalars, bold face letters (e.g. $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{w}$) to denote vectors, and upper case letters (e.g. $\mathcal{W}$, $\mathcal{X}$) to denote sets. The inner product between vectors $\boldsymbol{w}$ and $\boldsymbol{x}$ is denoted by $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$. The $[a]_+$ notation denotes a truncated function at 0, i.e. $[a]_+ = max(a, 0)$.

2.2 MLNs

An MLN consists of a set of weighted first-order clauses. It provides a way of softening first-order logic by making situations in which not all clauses are

satisfied less likely but not impossible [28]. More formally, let $\mathcal{X}$ be the set of all ground atoms, $\mathcal{C}$ be the set of all clauses in the MLN, w_i be the weight associated with clause $c_i \in \mathcal{C}$, $\mathcal{G}_{c_i}$ be the set of all possible groundings of clause c_i. Then the probability of a possible world $\mathbf{x}$ is defined as [28]:

$$P(\mathcal{X} = \mathbf{x}) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{c_i \in \mathcal{C}} w_i \sum_{g \in \mathcal{G}_{c_i}} g(\mathbf{x})\right) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{c_i \in \mathcal{C}} w_i n_i(\mathbf{x})\right)$$

where $g(\mathbf{x})$ is 1 if g is satisfied and 0 otherwise, $n_i(\mathbf{x}) = \sum_{g \in \mathcal{G}_{c_i}} g(\mathbf{x})$ is the number of true groundings of c_i in the possible world $\mathbf{x}$, and $\mathcal{Z} = \sum_{\mathbf{x} \in \mathcal{X}} \exp\left(\sum_{c_i \in \mathcal{C}} w_i n_i(\mathbf{x})\right)$ is the normalization constant.

In many applications, we know a priori which predicates provide evidence and which are used in queries, and the goal is to correctly predict query atoms given evidence atoms. If we partition the ground atoms in the domain into a set of evidence atoms $\mathcal{X}$ and a set of query atoms $\mathcal{Y}$, then the conditional probability of $\mathbf{y}$ given $\mathbf{x}$ is:

$$P(\mathcal{Y} = \mathbf{y} | \mathcal{X} = \mathbf{x}) = \frac{1}{\mathcal{Z}_{\mathbf{x}}} \exp\left(\sum_{c_i \in \mathcal{C}} w_i n_i(\mathbf{x}, \mathbf{y})\right)$$

where $n_i(\mathbf{x}, \mathbf{y})$ is the number of true groundings of c_i in the possible world ($\mathbf{x}$,$\mathbf{y}$) and $\mathcal{Z}_{\mathbf{x}} = \sum_{\mathbf{y} \in \mathcal{Y}} \exp\left(\sum_{c_i \in \mathcal{C}} w_i n_i(\mathbf{x}, \mathbf{y})\right)$ is the normalization constant.

2.3 Natural Language Field Segmentation

In this work, we look at an information extraction task, called field segmentation [10], a sample real-world problem where the data contains many, relatively small, structured examples in the form of short documents. A document is represented as a sequence of tokens, and the goal is to segment the text into fields, i.e. label each token in the document with a particular field. For example, when segmenting advertisements for apartment rentals [10], the goal is to segment each ad into fields such as *Features*, *Neighborhood*, *Rent*, *Contact*, etc.. Below are descriptions of some key predicates:

- $Token(string, position, docID)$: the token at a particular position in a document such as $Token(Entirely, P4, Ad001)$
- $InField(field, position, docID)$: the field label of the token at a particular position in a document such as $InField(Features, P4, Ad001)$
- $Next(position, position)$: the first position is next to the second, such as $Next(P01, P02)$

$InField$ is a target predicate, the rest are evidence predicates.

3 Online Max-Margin Structure and Parameter Learning

In this section, we describe our new online max-margin learning algorithm, OSL, for updating both the structure and parameters of an MLN. In each step, whenever the model makes wrong predictions on a given example, based on the wrongly predicted atoms the algorithm finds new clauses that discriminate the ground-truth possible world from the predicted one, then uses an adaptive subgradient method with l_1-regularization to update weights for both old and new clauses. Algorithm 1 gives the pseudocode for OSL. Lines 3 to 20 are pseudocode for structure learning and lines 21 to 35 are pseudocode for parameter learning.

3.1 Online Max-Margin Structure Learning with Mode-Guided Relational Pathfinding

Most existing structure learning algorithms for MLNs only consider ground-truth possible worlds and search for clauses that improve the likelihood of those possible worlds. However, these approaches may spend a lot of time exploring unhelpful clauses that are true in most possible worlds. Therefore, instead of only considering ground-truth possible worlds, OSL also takes into account the predicted possible worlds, i.e. the most probable possible worlds predicted by the current model. At each step, if the predicted possible world is different from the ground-truth one, then OSL focuses on where the two possible worlds differ and searches for clauses that differentiate them. This is related to the idea of using implicit negative examples in inductive logic programming (ILP) [34]. In this case, each ground-truth possible world plays the role of a positive example in traditional ILP. Making a closed world assumption [8], any possible world that differs from the ground-truth possible world is incorrect and can be considered as a negative example (the predicted possible world in this case). In addition, this follows the max-margin training criterion which focuses on discriminating the true label (the ground-truth possible world) from the most probable incorrect one (the predicted possible world) [33].

At each time step t, OSL receives an example $\mathbf{x}_t$, produces the predicted label $\mathbf{y}_t^P = \arg\max_{\mathbf{y}\in\mathbf{Y}} \langle \mathbf{w}_{\mathcal{C}}, \boldsymbol{n}_{\mathcal{C}}(\mathbf{x}_t, \mathbf{y}) \rangle$, then receives the true label $\mathbf{y}_t$. Given $\mathbf{y}_t$ and $\mathbf{y}_t^P$, in order to find clauses that separate $\mathbf{y}_t$ from $\mathbf{y}_t^P$, OSL first finds atoms that are in $\mathbf{y}_t$ but not in $\mathbf{y}_t^P$, $\Delta y_t = \mathbf{y}_t \setminus \mathbf{y}_t^P$. Then OSL searches the ground-truth possible world $(\mathbf{x}_t, \mathbf{y}_t)$ for clauses that are specific to the true ground atoms in Δy_t.

A simple way to find useful clauses specific to a set of atoms is to use relational pathfinding [27], which considers a relational example as a hypergraph with constants as nodes and true ground atoms as hyperedges connecting the nodes that are its arguments, and searches in the hypergraph for paths that connect the arguments of an input literal. A path of hyperedges corresponds to a conjunction of true ground atoms connected by their arguments and can be generalized into a first-order clause by variabilizing their arguments. Starting from a given atom, relational pathfinding searches for all paths connecting the arguments of the given atom. Therefore, relational pathfinding may be very

slow or even intractable when there are a large (exponential) number of paths. To speed up relational pathfinding, we use mode declarations [23] to constrain the search for paths. As defined in [23], mode declarations are a form of language bias to constrain the search for definite clauses. Since our goal is to use mode declarations for constraining the search space of paths, we introduce a new mode declaration: $modep(r, p)$ for paths. It has two components: a recall number r which is a positive integer, and an atom p whose arguments are place-makers. A place-maker is either '+' (input), '-' (output), or '.' (don't explore). The recall number r limits the number of appearances of the predicate p in a path to r. The place-maker restricts the search of relational pathfinding. Only paths connecting 'input' or 'output' nodes will be considered. A ground atom can only added to a path if one of its arguments has previously appeared as 'input' or 'output' arguments in the path and all of its 'input' arguments are 'output' arguments of previous atoms. Here are some examples of mode declarations for paths:

$$modep(2, Token(.,+,.)) \quad modep(1, Next(-,-)) \quad modep(2, InField(.,-,.)$$

The above mode declarations require that a legal path contains at most two ground atoms of each of the predicates $Token$ and $InField$ and one ground atom of the predicate $Next$. Moreover, the second argument of $Token$ is an 'input' argument; the second argument of $InField$ and all arguments of $Next$ are 'output' arguments. Note that, in this case, all 'input' and 'output' arguments are of type position. These 'input' and 'output' modes constrain that the position constants in atoms of $Token$ must appeared in some previous atoms of $Next$ or $InField$ in a path. From the graphical model perspective, these mode declarations restrict the search space to linear chain CRFs [31] since they constrain the search to paths connecting ground atoms of two consecutive tokens. It is easy to modify the mode declarations to search for more complicated structure. For example, if we increase the recall number of $Next$ to 2 and the recall number of $InField$ to 3, then the search space is constrained to second-order CRFs since they constrain the searches to paths connecting ground atoms of three consecutive tokens. If we add a new mode declaration $modep(1, LessThan(-,-))$ for the predicate $LessThan$, then the search space becomes skip-chain CRFs [31]. Algorithm 2 presents the pseudocode for efficiently constructing a hypergraph based on mode declarations by only constructing the hypergraph corresponding to input and output nodes. Algorithm 3 gives the pseudocode for mode-guided relational pathfinding, $ModeGuidedFindPaths$, on the constructed hypergraph. It is an extension of a variant of relational pathfinding presented in [14].[1] Starting from each true ground atom $r(c_1, ..., c_r) \in \Delta y_t$, it recursively adds to the path ground atoms or hyperedges that satisfy the mode declarations. Its search terminates when the path reaches a specified maximum length or when no new hyperedge can be added. The algorithm stores all the paths encountered during the search. Below is a sample path found by the algorithm:

[1] In this variant, a path does not need to connect arguments of the input atom. The only requirement is that any two consecutive atoms in a path must share at least one argument.

$$\{InField(Size, P29, Ad001), Token(And, P29, Ad001), Next(P29, P30),$$
$$Token(Spacious, P30, Ad001)\ InField(Size, P30, Ad001)\}$$

A standard way to generalize paths into first-order clauses is to replace each constant c_i in a conjunction with a variable v_i. However, for many tasks such as field segmentation, it is critical to have clauses that are specific to a particular constant. In order to create clauses with constants, we introduce mode declarations for creating clauses: $modec(p)$. This mode declaration has only one component which is an atom p whose arguments are either 'c' (constant) or 'v' (variable). Below are some examples of mode declarations for creating clauses:

$$modec(Token(c, v, v)) \quad modec(Next(v, v)) \quad modec(InField(c, v, v)$$

Based on these mode declarations, OSL variablizes all constants in a conjunction except those are declared as constants. Then OSL converts the conjunction of positive literals to clausal form since this is the form used in Alchemy.[2] In MLNs, a conjunction of positive literals with weight w is equivalent to a clause of negative literals with weight $-w$. Previous work [14,15] found that it is also useful to add other variants of the clause by flipping the signs of some literals in the clause. Currently, we only add one variant—a Horn version of the clause by only flipping the first literal, the one for which the model made a wrong prediction. In summary, for each path, OSL creates two type of clauses: one with all negative literals and one in which only the first literal is positive. For example, from the sample path above, OSL creates the following two clauses:

$$\neg InField(Size, p1, a) \vee \neg Token(And, p1, a) \vee \neg Next(p1, p2) \vee$$
$$\neg Token(Spacious, p2, a) \vee \neg InField(Size, p2, a)$$

$$InField(Size, p1, a) \vee \neg Token(And, p1, a) \vee \neg Next(p1, p2) \vee$$
$$\neg Token(Spacious, p2, a) \vee \neg InField(Size, p2, a)$$

Finally, for each new clause c, OSL computes the difference in the number of true groundings of c in the ground-truth possible world $(\mathbf{x}_t, \mathbf{y}_t)$ and the predicted possible world $(\mathbf{x}_t, \mathbf{y}_t^P)$, $\Delta n_c = n_c(\mathbf{x}_t, \mathbf{y}_t) - n_c(\mathbf{x}_t, \mathbf{y}_t^P)$. Then, only clauses whose difference in number of true groundings is greater than or equal to a predefined threshold *minCountDiff* will be added to the existing MLN. The smaller the value of *minCountDiff*, the more clauses will be added to the existing MLN at each step.

3.2 Online Max-Margin l_1-Regularized Weight Learning

The above online structure learner may introduce a lot of new clauses in each step, and some of them may not be useful in the long run. To address this issue, we use l_1-regularization which has a tendency to force parameters to zero

[2] The standard software for MLNs: alchemy.cs.washington.edu

Algorithm 1. OSL

Input: $\mathcal{C}$: initial clause set (can be empty)
mode: mode declaration for each predicate
$maxLen$: maximum number of hyperedges in a path
$minCountDiff$: minimum number of difference in true groundings for selecting new clauses
λ, η, δ: parameters for the l_1-regularization adaptive subgradient method
$\rho(\mathbf{y}, \mathbf{y}')$: label loss function

Note: Index H maps from each node γ_i to set of hyperedges $r(\gamma_1, ..., \gamma_i, ..., \gamma_n)$ containing γ_i
Paths is a set of paths, each path is a set of hyperedges

Initialize: $\mathbf{w}_\mathcal{C} = 0, \mathbf{g}_\mathcal{C} = 0, nc = |\mathcal{C}|$
for $i = 1$ **to** T **do**
  Receive an instance $\mathbf{x}_t$
  Predict $\mathbf{y}_t^P = \arg\max_{\mathbf{y} \in \mathbf{Y}} \langle \mathbf{w}_\mathcal{C}, \boldsymbol{n}_\mathcal{C}(\mathbf{x}_t, \mathbf{y}) \rangle$
  Receive the correct target $\mathbf{y}_t$
  Compute $\Delta y_t = \mathbf{y}_t \setminus \mathbf{y}_t^P$
  if $\Delta y_t \neq \emptyset$ **then**
    $H = CreateHG((\mathbf{x}_t, \mathbf{y}_t), mode)$
    $Paths = \emptyset$
    for each true atom $r(c_1, ..., c_r) \in \Delta y_t$ **do**
      $V = \emptyset$
      for each $c_i \in \{c_1, ..., c_r\}$ **do**
        if isInputOrOutputVar(c_i,mode) **then**
          $V = V \cup \{c_i\}$
        end if
      end for
      $ModeGuidedFindPaths(\{r(c_1, ..., c_r)\}, V, H, mode, maxLen, Paths)$
    end for
  end if
  $\mathcal{C}_{new} = CreateClauses(\mathcal{C}, Paths, mode)$
  Compute $\Delta \boldsymbol{n}_\mathcal{C}, \Delta \boldsymbol{n}_{\mathcal{C}_{new}}$:
    $\Delta \boldsymbol{n}_\mathcal{C} = \boldsymbol{n}_\mathcal{C}(\mathbf{x}_t, \mathbf{y}_t) - \boldsymbol{n}_\mathcal{C}(\mathbf{x}_t, \mathbf{y}_t^P)$
    $\Delta \boldsymbol{n}_{\mathcal{C}_{new}} = \boldsymbol{n}_{\mathcal{C}_{new}}(\mathbf{x}_t, \mathbf{y}_t) - \boldsymbol{n}_{\mathcal{C}_{new}}(\mathbf{x}_t, \mathbf{y}_t^P)$
  for $i = 1$ **to** $|\mathcal{C}|$ **do**
    $\mathbf{g}_{\mathcal{C},i} = \mathbf{g}_{\mathcal{C},i} + \Delta \boldsymbol{n}_{\mathcal{C},i} * \Delta \boldsymbol{n}_{\mathcal{C},i}$
    $\mathbf{w}_{\mathcal{C},i} = sign\left(\mathbf{w}_{\mathcal{C},i} + \frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathcal{C},i}}} \Delta \boldsymbol{n}_{\mathcal{C},i}\right) \left[\left|\mathbf{w}_{\mathcal{C},i} + \frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathcal{C},i}}} \Delta \boldsymbol{n}_{\mathcal{C},i}\right| - \frac{\lambda \eta}{\delta + \sqrt{\mathbf{g}_{\mathcal{C},i}}}\right]_+$
  end for
  for $i = 1$ **to** $|\mathcal{C}_{new}|$ **do**
    if $\Delta \boldsymbol{n}_{\mathcal{C}_{new,i}} \geq minCountDiffer$ **then**
      $\mathcal{C} = \mathcal{C} \cup \mathcal{C}_{new,i}$
      $nc = nc + 1$
      $\mathbf{g}_{\mathcal{C},nc} = \Delta \boldsymbol{n}_{\mathcal{C}_{new,i}} * \Delta \boldsymbol{n}_{\mathcal{C}_{new,i}}$
      $\mathbf{w}_{\mathcal{C},nc} = \left[\frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathcal{C},nc}}} (\Delta \boldsymbol{n}_{\mathcal{C}_{new,i}} - \lambda)\right]_+$
    end if
  end for
end for

Algorithm 2. CreateHG(D,mode)

Input: D: a relational example
mode: mode declaration file

for each constant c in D **do**
  $H[c] = \emptyset$
end for
for each true ground atom $r(c_1, ..., c_r) \in D$ **do**
  for each constant $c_i \in \{c_1, ..., c_r\}$ **do**
    if isInputOrOutputVar(c_i,mode) **then**
      $H[c_i] = H[c_i] \cup \{r(c_1, ..., c_r)\}$
    end if
  end for
end for
return H

Algorithm 3. $ModeGuidedFindPaths(CurrPath, V, H, mode, maxLen, Paths)$

if $|CurrPath| < maxLen$ **then**
  for each constant $c \in V$ **do**
    for each $r(c_1, ..., c_r) \in H[c]$ **do**
      if $canBeAdded(r(c_1, ..., c_r), CurrPath, mode) ==$ success **then**
        if $CurrPath \notin Paths$ **then**
          $CurrPath = CurrPath \cup \{r(c_1, ..., c_r)\}$
          $Paths = Paths \cup \{CurrPath\}$
          $V' = \emptyset$
          for each $c_i \in \{c_1, ..., c_r\}$ **do**
            if $c_i \notin V$ and isInputOrOutputVar(c_i,mode) **then**
              $V = V \cup \{c_i\}$
              $V' = V' \cup \{c_i\}$
            end if
          end for
          $ModeGuidedFindPaths(CurrPath, V, H, mode, maxLen, Paths)$
          $CurrPath = CurrPath \setminus \{r(c_1, ..., c_r)\}$
          $V = V \setminus V'$
        end if
      end if
    end for
  end for
end if

by strongly penalizing small terms [18]. We employ a state-of-the-art online l_1-regularization method—ADAGRAD_FB which is a l_1-regularized adaptive subgradient method using composite mirror-descent update [6]. At each time step t, it updates the weight vector as follows:

$$\mathbf{w}_{t+1,i} = sign\left(\mathbf{w}_{t,i} - \frac{\eta}{H_{t,ii}}\mathbf{g}_{t,i}\right)\left[\left|\mathbf{w}_{t,i} - \frac{\eta}{H_{t,ii}}\mathbf{g}_{t,i}\right| - \frac{\lambda\eta}{H_{t,ii}}\right]_+ \tag{1}$$

where λ is the regularization parameter, η is the learning rate, $\mathbf{g}_t$ is the subgradient of the loss function at step t, and $H_{t,ii} = \delta + ||\mathbf{g}_{1:t,i}||_2 = \delta + \sqrt{\sum_{j=1}^{t}(\mathbf{g}_{j,i})^2}$ ($\delta \geq 0$). Note that, ADAGRAD_FB assigns a different step size, $\frac{\eta}{H_{t,ii}}$, for each component of the weight vectors. Thus, besides the weights, ADAGRAD_FB also needs to retain the sum of the squared subgradients of each component.

From the equation 1, we can see that if a clause is not relevant to the current example (i.e. $g_{t,i} = 0$) then ADAGRAD_FB discounts its weight by $\frac{\lambda\eta}{H_{t,ii}}$. Thus, irrelevant clauses will be zeroed out in the long run.

Regarding the loss function, we use the prediction-based loss function l_{PL} [11], a simpler variant of the max-margin loss:

$$l_{PL}(\mathbf{w}_{\mathfrak{C}}, (\mathbf{x}_t, \mathbf{y}_t)) = \left[\rho(\mathbf{y}_t, \mathbf{y}_t^P) - \left\langle \mathbf{w}_{\mathfrak{C}}, \left(\mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t) - \mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t^P)\right)\right\rangle\right]_+$$

The subgradient of l_{PL} is:

$$\mathbf{g}_{PL} = \mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t^{PL}) - \mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t) = -\left[\mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t) - \mathbf{n}_{\mathfrak{C}}(\mathbf{x}_t, \mathbf{y}_t^{PL})\right] = -\Delta\mathbf{n}_{\mathfrak{C}}$$

Substituting the gradient into equation 1, we obtain the following formulae for updating the weights of old clauses:

$$\mathbf{g}_{\mathfrak{C},i} = \mathbf{g}_{\mathfrak{C},i} + (\Delta \boldsymbol{n}_{\mathfrak{C},\boldsymbol{i}})^2$$

$$\mathbf{w}_{\mathfrak{C},i} \leftarrow sign\left(\mathbf{w}_{\mathfrak{C},i} + \frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathfrak{C},i}}}\Delta \boldsymbol{n}_{\mathfrak{C},\boldsymbol{i}}\right)\left[\left|\mathbf{w}_{\mathfrak{C},i} + \frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathfrak{C},i}}}\Delta \boldsymbol{n}_{\mathfrak{C},\boldsymbol{i}}\right| - \frac{\lambda\eta}{\delta + \sqrt{\mathbf{g}_{\mathfrak{C},i}}}\right]_+$$

For new clauses, the update formulae are simpler since all the previous weights and gradients are zero:

$$\mathbf{g}_{\mathfrak{C},nc} = (\Delta \boldsymbol{n}_{\mathfrak{C}_{\boldsymbol{new,i}}})^2$$

$$\mathbf{w}_{\mathfrak{C},nc} = \left[\frac{\eta}{\delta + \sqrt{\mathbf{g}_{\mathfrak{C},nc}}}(\Delta \boldsymbol{n}_{\mathfrak{C}_{\boldsymbol{new,i}}} - \lambda)\right]_+$$

Lines 24 – 27 in Algorithm 1 are the pseudocode for updating the weights of existing clauses, and lines 28 – 35 are the pseudocode for selecting and setting weights for new clauses.

4 Experimental Evaluation

In this section, we conduct experiments to answer the following questions:

1. Starting from a given MLN, does OSL find new useful clauses that improve the predictive accuracy?
2. How well does OSL perform when starting from an empty knowledge base?
3. How does OSL compare to LSM, the state-of-the-art batch structure learner for MLNs [15] ?

4.1 Data

We ran experiments on two real world datasets for field segmentation: CiteSeer [17], a bibliographic citation dataset, and Craigslist [10], an advertisements dataset.

The CiteSeer dataset[3] contains 1,563 bibliographic citations. The dataset has four disjoint subsets consisting of citations in four different research areas. The task is to segment each citation into three fields: *Author*, *Title* and *Venue*.

The Craigslist dataset[4] consists of advertisements for apartment rentals posted on Craigslist. There are 8,767 ads in the dataset, but only 302 of them were labeled with 11 fields: *Available*, *Address*, *Contact*, *Features*, *Neighborhood*, *Photos*, *Rent*, *Restrictions*, *Roommates*, *Size*, and Utilities. The labeled ads are divided into 3 disjoint sets: training, development and test set. The number of ads in each set are 102, 100, and 100 respectively. We preprocessed the data using regular expressions to recognize numbers, dates, times, phone numbers, URLs, and email addresses.

4.2 Input MLNs

A standard model for sequence labeling tasks such as field segmentation is a linear-chain CRF [16]. Thus, we employ an initial MLN, named LC_0, which encodes a simple linear-chain CRF that uses only the current word as features:

$$
\begin{gathered}
Token(+t, p, c) \Rightarrow InField(+f, p, c) \\
Next(p1, p2) \wedge InField(+f1, p1, c) \Rightarrow InField(+f2, p2, c) \\
InField(f1, p, c) \wedge (f1! = f2) \Rightarrow \neg InField(f2, p, c).
\end{gathered}
$$

The plus notation indicates that the MLN contains an instance of the first clause for each (*token, field*) pair, and an instance of the second clause for each pair of fields. Thus, the first set of rules captures the correlation between tokens and fields, and the second set of rules represents the transitions between fields. The third rule constrains that the token at a position p can be part of at most one field.

For CiteSeer, we also use an existing MLN developed by Poon and Domingos [26], called the isolated segmentation model (ISM).[5] ISM is also a linear chain CRF but includes more features than the simple model above. Like LC_0, ISM also has rules that correlate the current words with field labels. For transition

[3] We used the versions created by Poon and Domingos [26], which can be found at `http://alchemy.cs.washington.edu/papers/poon07`

[4] `http://nlp.stanford.edu/~grenager/data/unsupie.tgz`

[5] `http://alchemy.cs.washington.edu/mlns/ie/ie_base.mln`

rules, ISM only captures transitions within fields and also takes into account punctuation as field boundaries:

$$Next(p1,p2) \wedge \neg HasPunc(p1,c) \wedge InField(+f,p1,c) \Rightarrow InField(+f,p2,c)$$

In addition, ISM also contains rules specific to the citation domain such as "the first two positions of a citation are usually in the author field", "initials tend to appear in either the author or the venue field". Most of those rules are features describing words that appear before or after the current word.

For Craigslist, previous work [10] found that it is only useful to capture the transitions within fields and take into account the field boundaries, so we create a version of ISM for Craigslist by removing clauses that are specific to the citation domain. Thus, the ISM MLN for Craiglist is a revised version of the LC_0 MLN. Therefore, we only ran experiments with ISM on Craigslist.

4.3 Methodology

To answer the questions above, we ran experiments with the following systems:

ADAGRAD_FB-LC_0: Use ADAGRAD_FB to learn weights for the LC_0 MLN.
OSL-M1-LC_0: Starting from the LC_0 MLN, this system runs a slow version of OSL where the parameter *minCountDiff* is set to 1, i.e. all clauses whose number of true groundings in true possible worlds is greater than those in predicted possible worlds will be selected.
OSL-M2-LC_0: Starting from the LC_0 MLN, this system runs a faster version of OSL where the parameter *minCountDiff* is set to 2.
ADAGRAD_FB-ISM: Use ADAGRAD_FB to learn weights for the ISM MLN.
OSL-M1-ISM: Like OSL-M1-LC_0, but starting from the ISM MLN.
OSL-M2-ISM: Like OSL-M2-LC_0, but starting from the ISM MLN.
OSL-M1-Empty: Like OSL-M1-LC_0, but starting from an empty MLN.
OSL-M2-Empty: Like OSL-M2-LC_0, but starting from an empty MLN.

Regarding label loss functions, we use Hamming (HM) loss which is the standard loss function for structured prediction [32,33].

For inference in training and testing, we used the exact MPE inference method based on Integer Linear Programming described by Huynh and Mooney [12]. For all systems, we ran one pass over the training set and used the average weight vector to predict on the test set. For Craigslist, we used the original split for training and test. For CiteSeer, we ran four-fold cross-validation (i.e. leave one topic out). The parameters λ, η, δ of ADAGRAD_FB were set to 0.001,1, and 1 respectively. For OSL, the mode declarations were set to constrain the search space of relational pathfinding to linear chain CRFs in order to make exact inference in training feasible; the maximum path length $maxLen$ was set to 4; the parameters λ, η, δ were set to the same values in ADAGRAD_FB. All the

Table 1. Experimental results for CiteSeer

Systems	Avg. F_1	Avg. train. time (min.)	Avg. num. of non-zero clauses
ADAGRAD_FB-LC_0	82.62 ± 2.12	10.40	2,896
OSL-M2-LC_0	92.05 ± 2.63	14.16	2,150
OSL-M1-LC_0	94.47 ± 2.04	163.17	9,395
ADAGRAD_FB-ISM	91.18 ± 3.82	11.20	1,250
OSL-M2-ISM	95.51 ± 2.07	12.93	1,548
OSL-M1-ISM	96.48 ± 1.72	148.98	8,476
OSL-M2-Empty	88.94 ± 3.96	23.18	650
OSL-M1-Empty	94.03 ± 2.62	257.26	15,212

Table 2. Experimental results for Craigslist

Systems	F_1	Train. time (min.)	Num. of non-zero clauses
ADAGRAD_FB-ISM	79.57	2.57	2,447
OSL-M2-ISM	77.26	3.88	2,817
OSL-M1-ISM	81.58	33.63	9,575
OSL-M2-Empty	55.28	17.64	1,311
OSL-M1-Empty	71.23	75.84	17,430

parameters are set based on the performance on the Craigslist development set. We used the same parameter values for CiteSeer.

Like previous work [26], to measure the performance of each system, we used F_1, the harmonic mean of the precision and recall, at the token level.

4.4 Results and Discussion

Table 1 shows the average F_1 with their standard deviations, average training times in minutes, and average number of non-zero clauses for CiteSeer. All results are averaged over the four folds. First, either starting from LC_0 or ISM, OSL is able to find new useful clauses that improve the F_1 scores. For LC_0, comparing to the system that only does weight learning, the fast version of OSL, OSL-M2, increases the average F_1 score by 9.4 points, from 82.62 to 92.05. The slow version of OSL, OSL-M1, further improves the average F_1 score to 94.47. For ISM, even though it is a well-developed MLN, OSL is still able to enhance it. The OSL-M1-ISM achieves the best average F_1 score, 96.48, which is 2 points higher than the current best F_1 score achieved by using a complex joint segmentation model that also uses information from matching multiple citations of the same paper [26]. Overall, this answers question 1 affirmatively. Additionally, the results for OSL-M2-Empty and OSL-M1-Empty shows that OSL also performs well when learning from scratch. OSL-M1 even finds a structure that is more accurate than ISM's. All differences in F_1 between OSL and ADAGRAD_FB are statistically significant according to a paired t-test ($p < 0.05$). Overall, this also answers question 2 affirmatively.

Regarding training time, OSL-M2 takes on average a few more minutes than systems that only do weight learning. However, OSL-M1 takes longer to train since including more new clauses results in longer time for constructing the ground network, running inference, and computing the number of true groundings. The last column of Table 1 shows the average number of non-zero clauses in the final MLNs learned by different systems. These numbers reflect the size of MLNs generated by different systems during training.

Table 2 shows the experimental results for Craigslist. The Craigslist segmentation task is much harder than CiteSeer's due to the huge variance in the context of different ads. As a result, most words only appear once or twice in the training set. Thus the most important rules are those that correlate words with fields and those capturing the regularity that consecutive words are usually in the same field, which are already in ISM. In addition, most rules only appear once in a document. That is why OSL-M2 is not able to find useful clauses, but OSL-M1 is able to find some useful clauses that improve the F_1 score of ISM from 79.57 to 81.58. OSL also gives some promising results when starting from an empty MLN.

To answer question 3, we ran LSM on CiteSeer and Craigslist but the MLNs returned by LSM result in huge ground networks that made weight learning infeasible even using online weight learning. The problem is that these natural language problems have a huge vocabulary of words. Thus, failing to restrict clauses to specific words results in a blow-up in the size of the ground network. However, LSM is currently not able to learn clauses with constants. It is unclear whether it is feasible to alter LSM to efficiently learn clauses with constants since such constants may need to be considered individually which dramatically increases the search space. This problem also holds for other existing MLN structure learners [13,21,1,14].

Below are some sample useful clauses found by OSL-M2-ISM on CiteSeer:

- If the current token is in the *Title* field and it is followed by a period then it is likely that the next token is in the *Venue* field.

$$InField(Ftitle, p1, c) \wedge FollowBy(p1, TPERIOD, c) \wedge Next(p1, p2) \Rightarrow InField(Fvenue, p2, c)$$

- If the next token is 'in' and it is in the *Venue* field, then the current token is likely in the *Title* field

$$Next(p1, p2) \wedge Token(Tin, p2, c) \wedge InField(Fvenue, p2, c) \Rightarrow InField(Ftitle, p1, c)$$

When starting from an empty knowledge base, OSL-M2 is able to discover the regularity that consecutive words are usually in the same field:

$$Next(p1, p2) \wedge InField(Fauthor, p1, c) \Rightarrow InField(Fauthor, p2, c)$$
$$Next(p1, p2) \wedge InField(Ftitle, p1, c) \Rightarrow InField(Ftitle, p2, c)$$
$$Next(p1, p2) \wedge InField(Fvenue, p1, c) \Rightarrow InField(Fvenue, p2, c)$$

5 Related Work

Our work is related to previous work on online feature selection for Markov Random Fields (MRFs) [25,35]. However, our work differs in two aspects. First, this previous work assumes all the training examples are available at the beginning and only the features are arriving online, while in our work both the examples and features (clauses) are arriving online. Second, in this previous work, all potential features are given upfront, while our approach induces new features from each example. Thus, our work is also related to previous work on feature induction for MRFs [3,19], but these are batch methods.

The idea of combining relational pathfinding with mode declarations has been used in previous work [24,5]. However, how they are used is different. In [24], mode declarations were used to transform a bottom clause into a directed hypergraph where relational pathfinding was used to find paths. Similarly, in [5], mode declarations were used to validate paths obtained from bottom clauses. Here, mode declarations are first used to reduce the search space to paths that contain 'input' and 'output' nodes. Then they are used to test whether an hyperedge can be added to an existing path. Finally, they are used to create clauses with constants.

6 Future Work

OSL, especially OSL-M1, currently adds many new clauses at each step, which significantly increases the computational cost. However, since OSL creates clauses from all the paths encountered in the search, some of the short clauses are subclauses of the long ones. So it may be better to only keep the long ones since they have more information. Second, OSL currently does not use clauses in the existing MLN to restrict the search space. So it would be useful to exploit this information. Finally, it would be interesting to apply OSL to other learning problems that involve data with many structured examples. For instance, other natural-language problems such as semantic role labeling or computer-vision problems such as scene understanding [7].

7 Conclusions

In this work, we present OSL, the first online structure learner for MLNs. At each step, OSL uses mode-guided relational pathfinding to find new clauses that fix the model's wrong predictions. Experimental results on field segmentation on two real-world datasets show that OSL is able to find useful new clauses that improve the predictive accuracies of well-developed MLNs and also learned effective MLNs from scratch.

Acknowledgments. The authors thank IBM for the free academic license of CPLEX. This research is supported by ARO MURI grant W911NF-08-1-0242 and by a gift from Microsoft Research. Most of the experiments were run on the Mastodon Cluster, provided by NSF Grant EIA-0303609. The first author also thanks the Vietnam Education Foundation (VEF) for its sponsorship.

References

1. Biba, M., Ferilli, S., Esposito, F.: Discriminative structure learning of markov logic networks. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 59–76. Springer, Heidelberg (2008)
2. Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 shared task: Semantic role labeling. In: Proc. of the 9th Conf. on Computational Natural Language Learning (CoNLL 2005), pp. 152–164 (2005)
3. Della Pietra, S., Della Pietra, V.J., Lafferty, J.D.: Inducing features of random fields. IEEE Trans. on Pattern Analysis and Machine Intelligence 19(4), 380–393 (1997)
4. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool Publishers, San Francisco (2009)
5. Duboc, A.L., Paes, A., Zaverucha, G.: Using the bottom clause and mode declarations on FOL theory revision from examples. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 91–106. Springer, Heidelberg (2008)
6. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Tech. rep., EECS Department, University of California, Berkeley (2010), http://www.cs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-24.html
7. Fei-Fei, L., Li, L.J.: What, Where and Who? Telling the Story of an Image by Activity Classification, Scene Recognition and Object Categorization. In: Computer Vision: Detection, Recognition and Reconstruction, pp. 157–171 (2010)
8. Genesereth, M.R., Nilsson, N.J.: Logical foundations of artificial intelligence. Morgan Kaufmann, San Francisco (1987)
9. Getoor, L., Taskar, B. (eds.): Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
10. Grenager, T., Klein, D., Manning, C.D.: Unsupervised learning of field segmentation models for information extraction. In: Proc. of the 43nd Annual Meeting of the Asso. for Computational Linguistics, ACL 2005 (2005)
11. Huynh, T.N., Mooney, R.J.: Online max-margin weight learning with Markov Logic Networks. In: Proc. of the 2011 SIAM Int. Conf. on Data Mining (SDM 2011), pp. 642–651 (2011)
12. Huynh, T.N., Mooney, R.J.: Max-Margin Weight Learning for Markov Logic Networks. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5781, pp. 564–579. Springer, Heidelberg (2009)
13. Kok, S., Domingos, P.: Learning the structure of Markov logic networks. In: ICML 2005 (2005)
14. Kok, S., Domingos, P.: Learning Markov logic network structure via hypergraph lifting. In: Proc. of 26th Int. Conf. on Machine Learning (ICML 2009), pp. 505–512 (2009)
15. Kok, S., Domingos, P.: Learning Markov logic networks using structural motifs. In: Proc. of 27th Int. Conf. on Machine Learning (ICML 2010), pp. 551–558 (2010)
16. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. of 18th Int. Conf. on Machine Learning (ICML 2001), pp. 282–289 (2001)
17. Lawrence, S., Giles, C.L., Bollacker, K.D.: Autonomous citation matching. In: Proc. of the 3rd Annual Conf. on Autonomous Agents, pp. 392–393 (1999)
18. Lee, S., Ganapathi, V., Koller, D.: Efficient structure learning of Markov networks using L_1-regularization. In: Adv. in Neu. Infor. Processing Systems (NIPS 2006), vol. 19, pp. 817–824 (2007)

19. McCallum, A.: Efficiently inducing features of conditional random fields. In: Proc. of 19th Conf. on Uncertainty in Artificial Intelligence (UAI 2003), pp. 403–410 (2003)
20. Mihalkova, L., Huynh, T., Mooney, R.J.: Mapping and revising Markov logic networks for transfer learning. In: Proc. of the 22nd Conf. on Artificial Intelligence (AAAI 2007), pp. 608–614 (2007)
21. Mihalkova, L., Mooney, R.J.: Bottom-up learning of Markov logic network structure. In: Proc. of 24th Int. Conf. on Machine Learning, ICML 2007 (2007)
22. Mihalkova, L., Mooney, R.J.: Learning to disambiguate search queries from short sessions. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 111–127. Springer, Heidelberg (2009)
23. Muggleton, S.: Inverse entailment and Progol. New Generation Computing 13, 245–286 (1995)
24. Ong, I.M., de Castro Dutra, I., Page, D., Costa, V.S.: Mode directed path finding. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 673–681. Springer, Heidelberg (2005)
25. Perkins, S., Theiler, J.: Online feature selection using grafting. In: Proc. of 20th Int. Conf. on Machine Learning (ICML 2003), pp. 592–599 (2003)
26. Poon, H., Domingos, P.: Joint inference in information extraction. In: Proc. of the 22nd Conf. on Artificial Intelligence (AAAI 2007), pp. 913–918 (2007)
27. Richards, B.L., Mooney, R.J.: Learning relations by pathfinding. In: Proc. of the 10th Nat. Conf. on Artificial Intelligence (AAAI 1992), pp. 50–55 (1992)
28. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62, 107–136 (2006)
29. Riedel, S., Meza-Ruiz, I.: Collective semantic role labelling with Markov logic. In: Proc. of the 12th Conf. on Computational Natural Language Learning (CoNLL 2008), pp. 193–197 (2008)
30. Slattery, S., Craven, M.: Combining statistical and relational methods for learning in hypertext domains. In: Page, D.L. (ed.) ILP 1998. LNCS, vol. 1446, pp. 38–52. Springer, Heidelberg (1998)
31. Sutton, C., McCallum, A.: An introduction to conditional random fields for relational learning. In: Getoor, L., Taskar, B. (eds.) Introduction to Statistical Relational Learning, pp. 93–127. MIT Press, Cambridge (2007)
32. Taskar, B., Guestrin, C., Koller, D.: Max-margin Markov networks. In: Adv. in Neu. Infor. Processing Systems, NIPS 2003, vol. 16 (2004)
33. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: Proc. of 21st Int. Conf. on Machine Learning (ICML 2004), pp. 104–112 (2004)
34. Zelle, J.M., Thompson, C.A., Califf, M.E., Mooney, R.J.: Inducing logic programs without explicit negative examples. In: Swierstra, S.D. (ed.) PLILP 1995. LNCS, vol. 982, pp. 403–416. Springer, Heidelberg (1995)
35. Zhu, J., Lao, N., Xing, E.P.: Grafting-light: fast, incremental feature selection and structure learning of Markov random fields. In: Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2010), pp. 303–312 (2010)

Fourier-Information Duality in the Identity Management Problem

Xiaoye Jiang[1], Jonathan Huang[2], and Leonidas Guibas[1]

[1] Stanford University, Stanford, CA, 94305, USA
[2] Carnegie Mellon University, Pittsburgh, PA, 15213, USA

Abstract. We compare two recently proposed approaches for representing probability distributions over the space of permutations in the context of multi-target tracking. We show that these two representations, the Fourier approximation and the information form approximation can both be viewed as low dimensional projections of a true distribution, but with respect to different metrics. We identify the strengths and weaknesses of each approximation, and propose an algorithm for converting between the two forms, allowing for a *hybrid* approach that draws on the strengths of both representations. We show experimental evidence that there are situations where hybrid algorithms are favorable.

1 Introduction

In this paper we consider the *identity management problem* which arises in a number of multi-target tracking scenarios in computer vision and robotics. Typical multi-target tracking systems maintain tracks of n people and the identity of the person corresponding to each track. A successful tracking system must reason in the face of noisy *evidence events*, in which an identity may be partially revealed to be at a particular track, as well as *mixing events*, in which identities can be confused when tracks cross paths.

To handle this uncertainty algorithmically, identity management is formalized mathematically as a filtering problem for identity-to-track associations, in which one must maintain a distribution over permutations. Since the space of permutations scales factorially in the number of tracked objects, n, however, it is not tractable to explicitly represent distributions over permutations for nontrivial n. Moreover, typical compact representations, such as graphical models, are not effective due to the mutual exclusivity constraints associated with permutations.

To efficiently represent and reason with such distributions, researchers have turned to a number of compact approximate representations. There are two competing methodologies in the identity management literature which have garnered the most attention in the last decade: the *Fourier theoretic* approach [6,7,11], and the *information theoretic* approach [14,17]. Cosmetically, both methods seem similar in spirit — the Fourier theoretic approach represents distributions over possible associations by maintaining marginal probabilities involving small subsets of objects, while the information theoretic approach represents similar terms, but working with unnormalized log-probabilities.

Despite progress made on both approaches over the last several years, there has been little work in unifying or even comparing the two approaches. In this paper we compare

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 97–113, 2011.

the Fourier and information approaches, drawing parallels between the two methods, and contrasting their strengths and weaknesses. The main contributions of our work is as follows:

1. Among the many parallels between the two representations, we identify an interesting duality between the two types of events (mixing and evidence) that must be processed during identity management. [6] showed that mixing events can be handled within the Fourier representation without increasing representational complexity, while evidence events always increase the representation complexity. We show that the opposite is true for the information form representation — that while evidence events can be handled without increased complexity, mixing events cannot be handled exactly without increasing representation complexity. We also make a connection between the two representations by viewing them as parametric representation of projected distributions with different metrics.
2. We explore the problem of converting between the Fourier and information theoretic representations and show that the conversion problem is #P-hard, but that due to recent advances in permanent approximation theory, approximate conversion is possible in polynomial time.
3. Using our algorithm for converting between the two forms, we propose a hybrid method that draws on the strengths of both representations and show experimental evidence that there are situations where hybrid algorithms are favorable.

2 Probabilistic Identity Management

In identity management, we are interested in maintaining a distribution over possible permutations which assign n identities to n tracks maintained by an internal tracker. We denote permutations as σ, where $\sigma(k)$ is the track belonging to the kth identity. Over time, the distribution over permutations in identity management is subject to change due to two causes: *mixing events* and *observation events*. In a mixing event, a subset of people can walk too closely together, leading to confusion about the identity-to-track associations for their respective tracks. This confusion is balanced by observation events, in which, for example, the color of an individual's clothing is captured by a sensor, giving information about his or her identity.

Uncertainty over permutations in identity management can be modeled with a hidden Markov model, where the joint probability of a sequence of latent permutations $(\sigma^{(1)} \ldots, \sigma^{(T)})$ and observed data $(z^{(1)}, \ldots, z^{(T)})$ factors as:

$$h(\sigma^{(1)}, \ldots, \sigma^{(T)}, z^{(1)}, \ldots, z^{(T)}) = h(z^{(1)}|\sigma^{(1)}) \cdot \prod_{t=1}^{T} h(z^{(t)}|\sigma^{(t)}) \cdot h(\sigma^{(t)}|\sigma^{(t-1)}).$$

We will refer to $h(\sigma^{(t)}|\sigma^{(t-1)})$ as the *mixing model*, which captures, for example, that tracks i and j swapped identities with some probability. We refer to $h(z^{(t)}|\sigma^{(t)})$ as the *observation model*, which captures, for example, the probability of observing a green blob given that Alice was at Track 1.

2.1 Inference Operations

There are two fundamental probabilistic inference operations that we focus on. The first is the *prediction/rollup* operation, which, given the distribution at time t, $h(\sigma^{(t)}|z^{(1)},\ldots,z^{(t)})$, and a mixing event, computes the distribution at the following timestep by multiplying by the mixing model and marginalizing over the permutation at the previous timestep:

$$h(\sigma^{(t_+1)}|z^1,\ldots,z^{(t)}) = \sum_{\pi\in S_n} h(\sigma^{(t+1)}|\sigma^{(t)}=\pi)\cdot h(\sigma^{(t)}=\pi|z^{(1)},\ldots,z^{(t)}).$$

The second is the *conditioning* operation, which, given a new observation $z^{(t+1)}$, performs a Bayesian update to compute the posterior distribution:

$$h(\sigma^{(t+1)}|z^1,\ldots,z^{(t+1)}) \propto \ell(z^{(t+1)}|\sigma^{(t+1)})\cdot h(\sigma^{(t+1)}|z^1,\ldots,z^{(t)}).$$

For explicit representations of the distribution $h(\sigma^{(t)})$, inference is intractable for all but very small n with running time complexities of $O((n!)^2)$ and $O(n!)$ for prediction/rollup and conditioning respectively. In this paper, we discuss two methods which have been proposed in recent years for compactly representing distributions over permutations and how these two inference operations can be performed efficiently with respect to each representation.

3 Two Dueling Representations

In this section we introduce the Fourier and information representations for distributions over permutations. In the simplest case, both representations maintain coefficients corresponding to the event that a single track j is associated with a single identity k, for all (track,identity) pairs j,k. Additionally, in both representations, one can also formulate generalizations which maintain coefficients corresponding to joint events that small subsets of identities map to small subsets of tracks. However, we show that with respect to the Fourier representation, the prediction/rollup step of inference is 'easy' in the sense that it can be performed efficiently and exactly, while the conditioning step of inference is 'difficult' since it can only be performed approximately. With respect to the information form representation, the roles are reversed, with prediction/rollup 'difficult' and conditioning 'easy'.

3.1 Fourier Domain Representation

The identity management problem was first introduced by Shin et al. [16], who proposed a representation based on collapsing the factorial sized distribution over permutations to just its *first-order marginals*, the n^2 marginal probabilities of the form:

$$H_{jk} = h(\sigma : \sigma(k) = j) = \sum_{\sigma\in S_n : \sigma(k)=j} h(\sigma).$$

The first-order marginals can be represented in a doubly stochastic matrix[1] (called a *belief matrix* in [16]). As an example, the matrix

$$H = \left[\begin{array}{c|ccc} & \text{Alice} & \text{Bob} & \text{Charlie} \\ \hline \text{Track 1} & 1/4 & 1/2 & 1/4 \\ \text{Track 2} & 3/8 & 3/8 & 1/4 \\ \text{Track 3} & 3/8 & 1/8 & 1/2 \end{array}\right].$$

By simply representing these first-order terms, it is already possible to make useful predictions. For example, we can predict the track at which the identity Alice is currently located, or predict the identity currently located at track 2.

The first-order marginal probabilities can be generalized to higher-order marginals which maintain, for example, the probability that a pair of tracks is jointly associated with a pair of identities. For example, we might be interested in the *second-order* probability that Alice and Bob are jointly in Tracks 1 and 2, respectively.

The reason for referring to these simple matrix-of-marginal type representations as 'Fourier' representations is due to the mathematical theory of generalized Fourier transforms for the symmetric group (see [3, 13, 15]). Just like the Fourier transform of a function on the real line can be separated into low and high frequency terms, a function over the symmetric group (the group of permutations) can be separated into low-order effects and higher-order effects. We remark that the Fourier coefficients of [6, 11] do not literally take the form of marginal probabilities but instead can be thought of as a set of coefficients which can be used to uniquely reconstruct the marginals. Loosely speaking, low-order marginal probabilities of a distribution can always be reconstructed using a subset of 'low-frequency' terms of its Fourier transform. Varying the maximum represented frequency yields a principled way for trading between accuracy and speed of inference.

Matrices of marginals can be viewed as a *compact summary* of a distribution over permutations, but they can additionally be viewed as an *approximation* to that distribution by applying the inverse Fourier transform to a truncated Fourier expansion. Given the first-order marginals H of a distribution, the approximate distribution is:

$$h(\sigma) = \frac{n-1}{n!}\mathrm{Tr}(H^T M_\sigma) - \frac{n-2}{n!}$$

where M_σ is the first-order *permutation matrix* associated with σ.[2] The above equation can be generalized to higher-order Fourier representations allowing for successively better approximations to the original distribution h.

[1] A doubly stochastic matrix has rows and columns which sum to 1. In the identity management setting, it reflects the constraint that every identity must map to *some* track, and that there is *some* identity on every track.

[2] Given a $\sigma \in S_n$, the permutation matrix associated with σ is defined as the $n \times n$ matrix M, with entries $M_{jk} = 1$ if $j = \sigma(k)$, 0 otherwise. This (first-order) permutation matrix can easily be generalized to higher-order permutation matrices whose nonzero entries represent assignments of tuples of identities $(k_1, \ldots, k_m)$ to tuples $(j_1, \ldots, j_m)$ of tracks.

Table 1. We compare common inference operations for the Fourier and information forms assuming the simplest case using a first-order representation, pairwise mixing, and first-order observations

Inference Operation	**Fourier (First Order)**		**Information Form (First Order)**	
	Accuracy	Complexity	Accuracy	Complexity
Prediction/Rollup	Exact	$\mathcal{O}(n)$	Approximate	$\mathcal{O}(n)$
Conditioning	Approximate	$\mathcal{O}(n^3)$	Exact	$\mathcal{O}(n)$
Normalization	Exact	$\mathcal{O}(n^2)$	Approximate	$\mathcal{O}(n^4 \log n)$
Maximization	Exact	$\mathcal{O}(n^3)$	Exact	$\mathcal{O}(n^3)$

3.2 Information Form Representation

Instead of representing the marginal probability that an identity k will be associated with track j, in the information form representation, one maintains a 'score' Ω_{jk} for each identity-track pair (k, j). The probability of a joint assignment of identities to tracks is parameterized as:

$$h(\sigma) = \frac{1}{Z_\Omega} \exp\left(\sum_{k=1}^{n} \Omega_{\sigma(k),k}\right) = \exp\Big(\mathrm{Tr}(\Omega^T M_\sigma)\Big),$$

where M_σ is the first-order *permutation matrix* associated with σ and Z_Ω is the normalizing constant. We observe that if we add a constant to every entry within a single row or single column of Ω, the distribution parameterized by Ω does not change. The entries of Ω are referred to as the *information coefficients* of the distribution P. Note that multiple settings of the information coefficient matrix Ω can correspond to the same distribution. For example, adding a constant c to any row or column of Ω does not change the distribution parameterized by Ω.

As with Fourier coefficients, it is possible to consider generalizations of the information form to higher order terms. For example, we can maintain a nonzero 'score' $\Omega'_{(j_1,j_2),(k_1,k_2)}$ where (j_1, j_2) denote a pair of tracks and (k_1, k_2) denote a pair of identities. Thus, in the information domain, the probability over permutations is parameterized as:

$$h(\sigma) = \frac{1}{Z_{\Omega'}} \exp\left(\sum_{k_1=1}^{n} \sum_{k_2 \neq k_1} \Omega_{(\sigma(k_1),\sigma(k_2)),(k_1,k_2)}\right) = \exp\Big(\mathrm{Tr}(\Omega^T M_\sigma)\Big),$$

where Ω' is a *second-order information coefficient matrix*.

3.3 Comparing the Two Representations

We now compare and contrast the two representations. Of particular interest are the probabilistic inference operations which are common in identity management. The challenge is how to perform these probabilistic operations using either the Fourier or information forms, exactly or approximately, in polynomial time. For simplicity, we will

restrict our focus to first order representations for both the Fourier and information domains. Additionally, we assume that mixing only occurs between a pair of tracks i and j at any given time, leading to the following simple mixing model in which one draws a permutation $\pi \sim m_{ij}(\pi)$, where:

$$m_{ij}(\pi) = \begin{cases} p & \text{if } \pi = id \\ 1-p & \text{if } \pi = (i,j) \\ 0 & \text{otherwise} \end{cases},$$

and sets $\sigma^{(t+1)} \leftarrow \pi \cdot \sigma_t$ (where $\cdot$ represents the composition of two permutations).

We also assume the simple observation model (employed in [6,11]) which assumes that we get observations z of the form: 'track j is color r'. The probability of seeing color r at track j given an identity-to-track association σ is

$$\ell(\sigma) = \text{Prob}(\text{track } j \text{ is color } r|\sigma) = \alpha_{\sigma^{-1}(j),r},$$

where $\sum_r \alpha_{\sigma^{-1}(j),r} = 1$. The likelihood model parameters α can be constructed based on prior knowledge of color profiles of the moving targets [6].

For a tabular summary of the inference operations considered in this section, we refer the reader to Table 1.

Prediction/Rollup. In general, the pairwise mixing models considered in this paper can be thought of as a special case of random walk transitions over a group, which assume that $\sigma^{(t+1)}$ is generated from $\sigma^{(t)}$ by drawing a random permutation $\pi^{(t)}$ from some distribution $m^{(t)}$ and setting $\sigma^{(t+1)} = \pi^{(t)}\sigma^{(t)}$. The permutation $\pi^{(t)}$ represents a random identity permutation that might occur among tracks when they get close to each other (what we call a *mixing event*).

The motivation behind the random walk transition model is that it allows us to write the prediction/rollup operation as a *convolution* of distributions, and as a result the familiar *convolution theorem* of Fourier analysis holds. Below we state the convolution theorem for the special case of first order Fourier representations, but a more general statement can be found in, for example, [6].

Proposition 1 (Convolution theorem). *Let $M^{(t)}$ be the first order matrix of marginals for the distribution $m^{(t)}$ and $H^{(t)}$ be the first order matrix for $h(\sigma^{(t)}|z^{(1)},\ldots,z^{(t)})$. The first order matrix for the distribution after the prediction step, $h(\sigma^{(t+1)}|z^{(1)},\ldots,z^{(t)})$ is:*

$$H^{(t+1)} = M^{(t)} \cdot H^{(t)},$$

where the operation on the right side is matrix multiplication.

Prediction/rollup in the Fourier domain is *exact* in the sense that first order marginals for timestep $t+1$ can be computed exactly from first order marginals at timestep t. In contrast the same operation cannot be performed exactly with respect to information form coefficients and in particular, we argue that, if the distribution $h(\sigma^{(t)})$ can be represented with first order information form coefficients, then under pairwise mixing, second order information form coefficients are necessary and sufficient for representing the distribution $h(\sigma^{(t+1)})$.

Proposition 2. *Let $\Omega^{(t)}$ be the first order information coefficient matrix for the distribution $h(\sigma^{(t)}|z^{(1)},\ldots,z^{(t)})$. There exists a second order information coefficient matrix $\Omega^{(t+1)}$ which exactly parameterizes the distribution obtained by the prediction/rollup step $h(\sigma^{(t+1)}|z^{(1)},\ldots,z^{(t)})$ in the information domain.*

Proof. Given information coefficients Ω which parametrize $h(\sigma^{(t)}|z^{(1)},\ldots,z^{(t)})$, we argue that there exists an $n(n-1)$-by-$n(n-1)$ 2nd order information coefficient matrix Ω' which exactly parametrize $h(\sigma^{(t+1)}|z^{(1)},\ldots,z^{(t)})$. To see this, suppose that track k_1 and k_2 mixed up, then the distribution after the rollup operation evaluated on σ would be proportional to

$$p\exp\left(\mathrm{Tr}(\Omega^T M_\sigma)\right)+(1-p)\exp\left(\mathrm{Tr}(\Omega^T M_{\pi\sigma})\right).$$

In such a expression, any entries in Ω that does not lie in row k_1 or k_2 is still an additive term in the logarithmic space for characterizing the posterior.

For entries that lie in either row k_1 or k_2, we need to form $n(n-1)$ 2nd order information coefficients $\Omega_{(\sigma(k_1),\sigma(k_2)),(k_1,k_2)}$. With those coefficients, we can represent the posterior distribution evaluated on σ using logarithmic likelihoods $\Omega_{(\sigma(k_1),\sigma(k_2)),(k_1,k_2)}$ together with $\Omega_{\sigma(k),k}$, where $k\neq k_1,k_2$.

It turns out that the above logarithmic likelihoods can be combined together into a second order information matrix. This is because the representation theory applies to the logarithmic space of the information form representation. □

Instead of increasing the size of the representation at each timestep, a sensible approximation is to compute a projection of $h(\sigma^{(t+1)})$ to the space of distributions which can be represented in first-order information form. Schumitsch et al. [14] proposed the following update:

$$\Omega^{(t+1)}=\log\left(M^{(t)}\cdot\exp(\Omega^{(t)})\right)$$

which they showed worked well in practice. The exponential and logarithmic functions in the formula refer to elementwise operations rather than matrixwise operations.

Conditioning. In contrast with the ease of prediction/rollup operations, conditioning a distribution in the Fourier domain is more complex and increases the size of the representation.

Proposition 3 (Kronecker conditioning [6]). *Let $H^{(t+1)}$ be the first order matrix of marginals for the distribution $h(\sigma^{(t+1)}|z^{(1)},\ldots,z^{(t)})$, then there exists a second order matrix of marginals which exactly parametrize the distribution obtained by the conditioning step $h(\sigma^{(t+1)}|z^{(1)},\ldots,z^{(t+1)})$ in the Fourier domain.*

Using information coefficients, however, conditioning can be performed exactly, and takes a particularly simple and efficient form (that of a local addition) which does *not* increase the representation complexity.

Proposition 4 (Schumitsch et al. [14]). *If $h(\sigma)\propto\exp\left(Tr(\Omega^T M_\sigma)\right)$, then the update is of the form*

$$\Omega_{jk}\leftarrow\Omega_{jk}+\log\alpha_{k,r}.$$

where $k=\sigma^{-1}(j)$. The complexity of this update is $\mathcal{O}(n)$.

Normalization and Maximization. Normalization is a major inference operation and appears, for example, as a subroutine of the conditioning and marginalization operations, i.e., computing $\sum_\sigma \ell(\cdot|\sigma)h(\sigma|\cdots)$ or $\sum_\sigma h(\sigma)$. In the Fourier domain, normalization is 'free' since the *zeroth-order* marginal is exactly the normalization constant $Z = \sum_\sigma h(\sigma)$. Thus with respect to the irreducible Fourier coefficients of [6,11], normalization can be performed by dividing all Fourier coefficients by the lowest-frequency coefficient. Alternatively, if the matrix of marginals, H, is represented explicitly, the normalization constant Z is simply the sum across any row or column of H. One can then normalize by scaling every entry of H by Z.

It may be somewhat surprising to realize that the normalization problem is provably hard in the information domain since the probability of a joint assignment may at first glance seem to factorize as:

$$h(\sigma) \propto \prod_{k=1}^{n} w_{k,\sigma(k)} = \exp(\sum_k \Omega_{\sigma(k),k}),$$

which would allow one to factor the normalization problem into tractable pieces. However, due to mutual exclusivity constraints which disallow identities from mapping to the same track, probabilistic independence is not present. Instead, the normalization constant, $Z = \sum_{\sigma \in S_n} \prod_k W_{k,\sigma(k)}$, is exactly the matrix permanent of $W = \exp(\Omega)$, whose computation is #P-complete (even for binary matrices). We have:

Proposition 5. *Computing the normalization constant of the information form paramterization is #P-complete.*

We remark that despite the dramatic differences with respect to normalization, computing the permutation which is assigned the maximum probability under h (instead of summing over h) reduces to the same problem for both the Fourier and information forms due to the fact that the exponential is a monotonic function. In the first-order case, for example, one must compute $\arg\max_\sigma \operatorname{Tr}\left(H^T M_\sigma\right)$ (see Equation 3.1), which can be efficiently solved using either linear programming or a number of other combinatorial algorithms.

Both Forms are Low-Dimensional Projections. Since the Fourier transform is linear and orthogonal [3], the Fourier approximation of a distribution h over permutations can be thought of as an ℓ_2 projection of h onto a low-frequency Fourier basis V which can be interpreted as affine marginal constraints. This projection is associated with the following *Pythagorean theorem*, which says that if g is any function lying in the span of V, then $\|g-h\|_{\ell_2}^2 = \|g-h'\|_{\ell_2}^2 + \|h'-h\|_{\ell_2}^2$, where h' is the Fourier projection of h onto the span of V.

The information form representation can be thought of, on the other hand, as an information projection of h to the same low-frequency Fourier subspace V using the *KL-divergence* metric. Recall that the KL-divergence, also known as the *relative entropy* is defined as $D(q||h) = \sum_\sigma q(\sigma) \log \frac{q(\sigma)}{h(\sigma)}$. Given a doubly stochastic matrix H of first order marginals, the information projection (IP) can be formulated as follows:

$$(IP)\ \min_q \quad \sum_\sigma q(\sigma)\log\frac{q(\sigma)}{h(\sigma)} \qquad (ME)\quad \min_q \quad \sum_\sigma q(\sigma)\log q(\sigma)$$
$$\text{s.t.} \quad \sum_\sigma q(\sigma)M_\sigma = H \qquad\qquad \text{s.t.} \quad \sum_\sigma q(\sigma)M_\sigma = H$$
$$q(\sigma)\geq 0, \forall\sigma \qquad\qquad q(\sigma)\geq 0, \forall\sigma$$

In the special case, where the distribution h to be projected is uniform, i.e., we have no prior knowledge, then the information projection problem becomes the maximum entropy (ME) problem. The objective in (ME) coincides with the maximum entropy principle in Bayesian probability, where the information entropy of a distribution q over S_n is $H[q] = -\sum_\sigma q(\sigma)\log q(\sigma)$. The maximum entropy distribution can be thought of as the least biased distribution encoding some given information (about low-order marginals in our case). We remark that the normalization constraint $\sum_\sigma q(\sigma) = 1$ is implicitly contained in the first constraint, $\sum_\sigma q(\sigma)M_\sigma = H$.

The following result (see proof of Proposition 7) shows that the solution to maximum entropy problem *must* be parametrizable as an information form distribution:

Proposition 6. *The solution to* (IP) *is guaranteed to take the form* $h(\sigma)\exp\left(Tr(\Omega^T M_\sigma)\right)$ *while the solution to* (ME) *is guaranteed to take the form* $q(\sigma)\propto\exp\left(Tr(\Omega^T M_\sigma)\right)$. *The* Pythagorean theorem *holds: if* g *is any function that satisfies the marginal constraints, then* $D(g||h) = D(g||h') + D(h'||h)$, *where* h' *is the information projection of* h.

3.4 Discussion

As we have shown, both the Fourier and information forms can be thought of as methods for approximating distributions over permutations via a low-dimensional projection. However, we have also argued that each method has their own respective advantages and disadvantages with respect to the two inference operations of prediction/rollup and conditioning. While prediction/rollup updates, which increase the information entropy of the maintained distribution, can be performed exactly with respect to a Fourier representation, conditioning updates, which typically decrease the entropy, can be performed exactly with respect to an information form representation. As a result, Fourier representations are typically much more suitable for modeling problems with high uncertainty, while information form representations are more suitable for problems with low uncertainty. In Section 7, we will validate these claims with experiments.

4 Representation Conversion

In this section we show a natural method for converting between the two representations. Since the two representations do not describe the same space of functions, conversion can only be approximate. We show in particular that much like the normalization problem which we discussed in the previous section, converting between the two representations requires solving the *matrix permanent problem*.

4.1 From Information Coefficients to Fourier Coefficients

We first consider the problem of estimating low-order marginals from information coefficients. Given the information coefficients Ω, we can compute the first order marginal

probability that identity k maps to j, H_{jk}, by conditioning on $\sigma(k) = j$, then normalizing. Note that the posterior after conditioning can also be written in information form and that the normalization operation corresponds to taking the permanent of the information matrix of the posterior distribution. We have:

$$H_{jk} = \sum_{\sigma:\sigma(k)=j} h(\sigma) = \frac{\exp(\Omega_{jk})\mathrm{perm}(\exp(\hat{\Omega}_{jk}))}{\mathrm{perm}(\exp(\Omega))}.$$

Here $\hat{\Omega}_{jk}$ denotes the $n-1$ by $n-1$ submatrix of Ω with the j'th row and k'th column removed. The matrix $\exp(\hat{\Omega}_{jk})$ denotes component-wise exponentials rather than matrix exponentials. We therefore conclude that to convert from information coefficients to Fourier coefficients, one must compute matrix permanents.

4.2 From Fourier Coefficients to Information Coefficients

We now discuss the opposite conversion from Fourier coefficients to Information coefficients, for which we take the maximum entropy approach described in the previous section (problem (ME)). Given, say, the first-order marginal probabilities, we are interested in computing the maximum entropy distribution consistent with the given marginals, which we argued can be parameterized in information form. We now turn to the problem of algorithmically optimizing the entropy with respect to low-order constraints. Our approach is to solve the dual problem [1]:

Proposition 7. *The dual problem of* (ME) *is:*

$$\max_{Y} \; Tr\Big(Y^T H\Big) - \sum_{\sigma} \exp\Big(Tr(Y^T M_\sigma) - 1\Big),$$
$$s.t. \; Y \leq 0.$$

Proof. The Lagrangian for (ME) is given by:

$$\sum_{\sigma} q(\sigma)\log q(\sigma) - \sum_{\sigma} s_\sigma q(\sigma) - \mathrm{Tr}\Big(Y^T(\sum_{\sigma} q(\sigma)M_\sigma - H)\Big),$$

where s_σ and Y are dual variables associated with the constraint $q(\sigma) \geq 0$, and $\sum_\sigma q(\sigma)M_\sigma = Q$. The KKT conditions tell us that for (ME):

$$1 + \log q(\sigma) - s_\sigma - \mathrm{Tr}\Big(Y^T M_\sigma\Big) = 0.$$

Assuming all $q(\sigma) > 0$, which gives us $s_\sigma = 0$ becuase of the dual complementary condition, we have

$$q(\sigma) = \exp\Big(\mathrm{Tr}(Y^T M_\sigma) - 1\Big) = \exp\Big(\Omega^T M_\sigma\Big)$$

where $\Omega = Y - 1/n$. Thus implies that the distribution q is completely characterized by n^2 information coefficients Ω. So the dual objective of (ME) is therefore

$$\mathrm{Tr}\Big(Y^T H\Big) - \sum_{\sigma} \exp\Big(\mathrm{Tr}(Y^T M_\sigma) - 1\Big). \qquad \square$$

Gradient Based Optimization for the Maximum Entropy Problem. We now give a simple gradient descent algorithm to find the solution of the dual problem. Note that the gradient of the objective function is given by the matrix

$$G(Y) = H_{jk} - \sum_{\sigma} \exp\left(\mathrm{Tr}(Y^T M_\sigma)\right) - 1)(M_\sigma)_{jk} = H_{jk} - \exp(Y_{jk} - 1)\mathrm{perm}(\exp(\hat{Y}_{jk})).$$

Thus we can have a simple gradient descent algorithm, where at each iteration we find an optimal step length α such that the objective function values is improved, i.e.,

$$\mathrm{Tr}\left((Y + \alpha G(Y)^T H\right) - \sum_{\sigma} \exp\left(\mathrm{Tr}((Y + \alpha G(Y))^T M_\sigma) - 1\right) > \mathrm{Tr}\left(Y^T H\right) - \sum_{\sigma} \exp\left(\mathrm{Tr}(Y^T M_\sigma) - 1\right),$$

while the feasibility $Y + \alpha G(Y) \leq 0$ is still maintained. We note that the estimation of the gradient involves estimating the matrix permanent which we now discuss.

The pseudocode for the algorithm is given below.

4.3 Computation of the Matrix Permanent

We have shown that the problems of converting between the two above representations both require one to solve the matrix permanent problem, one of the prototypically #*P*-complete problems (even when all of the entries are binary [9]). The fastest known general exact algorithm is due to Ryser [12] based on the inclusion-exclusion formula.

Algorithm 1. Computing Information Coefficients Ω from Marginals Q

$Y \Leftarrow 0$
while $\|G(Y)\| \geq \varepsilon$ **do**
  Find an optimal step length α
  $Y \Leftarrow Y + \alpha G(Y)$
end while
$\Omega \Leftarrow Y$

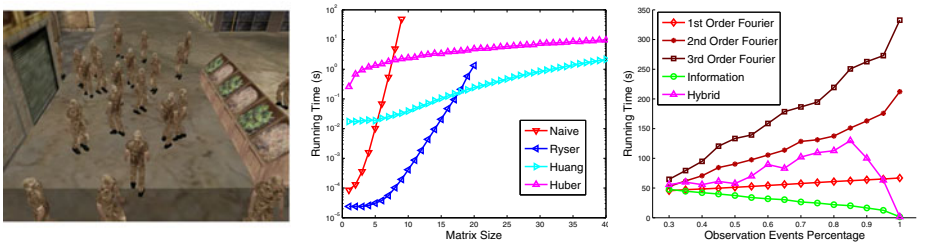
In some special cases, polynomial time algorithms exist for estimating the matrix permanent (e.g., for planar graphs [10]), but we have not found any such special cases to be applicable for general identity management problems.

When the entries of the matrix are non-negative, which is true in our setting there is an FPRAS (fully polynomial-time randomized approximation scheme) for approximating the permanent in probabilistic polynomial time [8,9].

Finally, the fastest approximation that we are aware of is based on the Bethe free energy approximation [5,18,19] which frames the permanent problem as an inference problem in a graphical model, which can then be solved using loopy belief propagation.

5 A Hybrid Approach for Identity Management

Using the conversion algorithms presented in the previous section, we now present a *hybrid identity management approach* in which we switch between the Fourier and



(a) (b) (c)

Fig. 1. (a) A view of the simulated data. (b) Running time comparison of different approaches in computing matrix permanent. (c) Comparing the running time of the three approaches.

information form domains depending on which domain is more convenient for certain inference operations. There are several issues that one must consider in designing a scheme for switching between the two domains. In this section, we present three simple switching strategies which we compare experimentally in Section 7.

We have argued that to handle mixing events, it is better to use a Fourier representation and that to handle evidence events, it is better to use the information form representation. A simple switching strategy (which we call the *myopic switching* scheme) thus *always* switches to either the Fourier or information form domain depending on whether it must perform a prediction/rollup operation or a conditioning operation.

In a similar spirit, we can also consider a *smoothness based switching* scheme, in which we switch based on the diffuseness of the distribution. In our implementation, we consider a heuristic in which we switch to a Fourier representation whenever the first-order matrix of marginals is within ε of a uniform matrix with respect to the Frobenius norm. Similarly, we switch to the information form representation whenever the first-order matrix comes within ε of some delta distribution.

What both the myopic and smoothness based approaches ignore, however, is the computational cost of switching between representations. To minimize this switching cost, we finally propose the *lagged block switching* scheme in which switching is only allowed to happen every k timeslices, where k is a parameter set by the user. In lagged block switching, we allow the identity management algorithm to lag the incoming data by k timesteps and therefore it can look ahead to see whether there are more mixing events or evidence events in the next k timesteps. As with myopic switching, the algorithm switches to Fourier if there are a majority of mixing events, and switches to information form if there are a majority of evidence events. After potentially switching, the algorithm processes the next k timesteps sequentially.

6 An Adaptive Approach for Identity Management

There are two extremal cases in the identity management problem: if we are completely uncertain about the assignment of target identities to tracks, then we have a uniform distribution over permutations, this smooth distribution can be represented compactly with Fourier coefficients; at the limit when we know the location of every identity, our distribution becomes very peaked, and we can use information coefficients to represent such

a distribution compactly. In a real tracking scenario, we can pull highly certain or uncertain groups of targets out of a global Fourier or information representation and represent them separately, so that the problem breaks up into independent subproblems [7]. We now propose a method based on exploiting probabilistic independence of distributions over permutations, which can achieve significantly improved scalability.

Due to the mutual exclusivity constraints associated with permutations, we say the distribution $h(\sigma)$ has a independence factorization if there exists a subset X of identities and a subset Y of tracks, and also their corresponding complement subsets $\bar{X}$ and $\bar{Y}$, such that $h(\sigma)$ can be factorized into a product of two distributions over all mappings between X and Y and all mappings between $\bar{X}$ and $\bar{Y}$.

It turns out that whenever probabilistic independence holds, then both first order Fourier coefficients and information coefficients can be rendered block diagonal under an appropriate reordering of the rows and columns [7]. Since X and Y are unknown, our task is to find permutations of the rows and columns of the first order Fourier or information coefficients to obtain a block diagonal matrix. Viewing such a matrix as a set of edge weights on a bipartite graph between identities and tracks, we can approach the detection step as a biclustering problem with an extra balance constraint forcing $|X| = |Y|$. In practice, we use a cubic time SVD-based technique presented in [20] which finds bipartite graph partitions optimizing the normalized cut measure modified to satisfy the balance constraint. We note that such bipartite graph partitioning problems can be approached using either the ℓ_2 metric [20] or KL-divergence metric [2].

7 Experiments

In this section, we perform several experiments to compare the Fourier approach, information approach and the proposed hybrid approach. We use the Delta3D game engine to generate simulated crowds of up to 50 moving targets which walk around in an outdoor market [4]; Figure 1-(a) depicts a snapshot view of the simulated crowd. Such a simulation approach allows us to obtain accurate ground truth for large crowds than would be feasible in a typical physical testbed. The data contains interesting movement patterns and we can extract mixing and observation events directly from the data. We log a mixing event whenever two targets get within some distance of each other and an observation event whenever one target is separated from all the other targets for some distance. The percentages of observation events can be controlled by adjusting those distance parameters. We measure tracking errors using the fraction of mislabeled target identities over the tracks.

We first run an experiment for testing the running time performance of different algorithms in estimating the matrix permanent. As shown in Figure 1-(b), we generate random matrices and compare the running time of the four approaches: the naive method which sums up all products of matrix elements that lie in different rows and columns, the fastest known exact algorithm by Ryser [12], the Monte Carlo sampling algorithm by Huber et al. [8], and the loopy belief propagation algorithm by Huang et al. [5]. The naive approach has a super-exponential complexity and the Ryser's formula has an exponential complexity, thus, they scale poorly as the matrix size grows; On the other hand, the two randomized approximate algorithms have much better running time

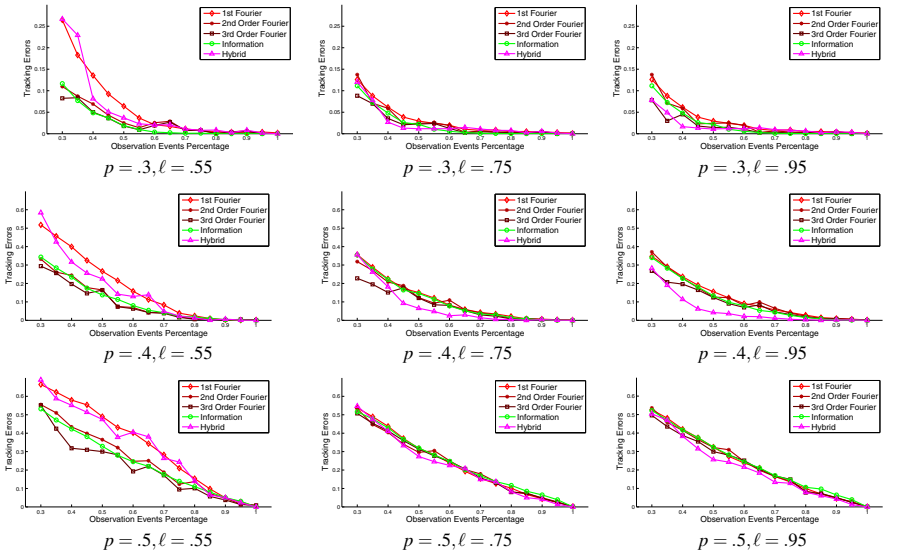


Fig. 2. Comparing tracking accuracy of the three approaches with different parameters

performance than the exact algorithms. In the hybrid algorithm, we use Monte Carlo sampling algorithms [8]. for estimating the matrix permanent.

In our experiments, we can control two sets of parameters which determine the tracking quality, one is the swapping probability — if we can keep track of who is who when two targets mix with high probability during the prediction/rollup operation, we can achieve better tracking performance; the other is the likelihood function, if the likelihood for observing the identity of a target is high, then conditioning step can resolve the ambiguities better. We set up nine cases to explore the tracking accuracy with different swapping probability and likelihood function parameters. As depicted in Figure 2, the probability p characterizing confusions of the mixing events grows larger from left to right, and the likelihood ℓ for observing target identity grows larger from top to bottom. We can get better tracking accuracy if p is small or ℓ is large.

From Figure 2, we can see that the information approach outperforms the Fourier approach in most cases, while the Fourier approach can beat the information approach only slightly in some cases, e.g., the case $p = .5, \ell = .75$ where the mixings are quite confusing. The tracking accuracy can be improved if we incorporate high order Fourier coefficients. We can achieve better performances in lots of cases if we use the hybrid approach, whose tracking accuracy are comparable to the 2nd order or even 3rd order Fourier approach. The running time for those approaches are shown in Figure 1-(c). In general, the Fourier approach has a fundamental trade-off between tracking complexity in terms of the number of coefficients used and the tracking accuracy: we can improve tracking accuracy by using more coefficients. The hybrid approach makes a good balance which can improve tracking accuracy when there are observation events that confirm the target identities (large ℓ) with moderate running time. We can see that the running time for the hybrid approach is strictly less than the second order Fourier

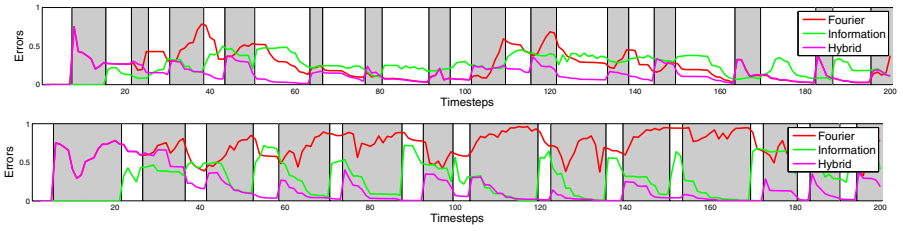
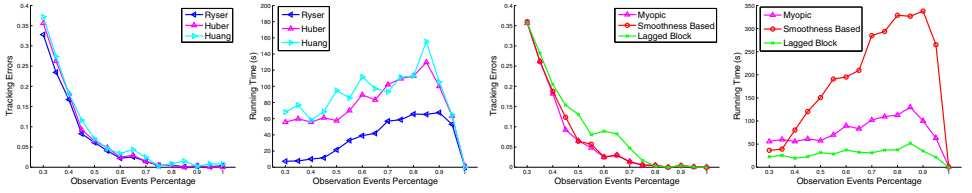


Fig. 3. Compare the errors in distribution of the three approaches. The white intervals denote the rollup steps and the grey intervals denote the conditioning steps.



(a) (b) (c) (d)

Fig. 4. (a,b) Tracking accuracy and running time of the hybrid approach with different algorithms for estimating matrix permanent. (c,d) Tracking accuracy and running time of the hybrid approach with different rules for switching.

approach. This is because the complexity for the conditioning step in the Fourier domain is very expensive if we use high order Fourier coefficients.

We also compare the errors of approximating the distribution over permutations of the three approaches (see Figure 3). It turns out that the Fourier approach decrease (increase) the errors during the rollup (conditioning) steps, while the information approach decrease (increase) the errors during the conditioning (rollup) steps. However, if we use the hybrid approach, we can always keep the errors at a much lower level.

We also compare the tracking accuracy and runing time of the hybrid approach by varing the algorithms for estimating the matrix permanent, as well as varying the strategies for switching between two domains. Specifically, we compare Ryser, Huber, and the LBP algorithms for estimating matrix permanents. The tracking accuracy of those approaches does not differ too much. However, the two approximation algorithms (by Huber and the loopy belief propagation method) have longer running times for the small scale experiments because it takes longer time to converge in solving the maximum entropy problem (see Figure 4-(a,b)). We also evaluate our three different switching strategies for the hybrid approach. Compared with the smoothness based switching strategy which switches 38 times, the lagged block strategy switches between two domains 91 times among the 1000 timesteps while can not improve the tracking accuracy too much and take very long running time. The myopic strategy suffers a little on the tracking accuracy while the running time can be improved because there are only 20 times of switchings (see Figure 4-(c,d)).

We finally evaluate the performance of the adaptive approach. As depicted in Figure 5, the tracking accuracy for the adaptive approach is comparable to the

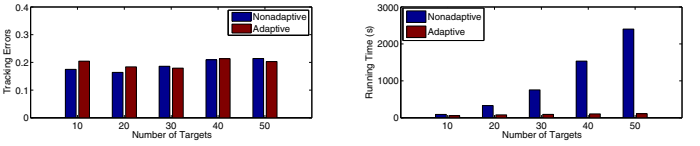


Fig. 5. Tracking accuracy and running time of the adaptive approach compared with the nonadaptive approach

nonadaptive approach, while the running time can always be controled using the adaptive approach. In particular, the tracking accuracy for the adaptive approach is often worse than the nonadaptive approach when the number of targets is small because it is usually difficult to factorize the problem in those cases. When the number of targets is larger, however, the benefit of adaptive approach becomes more evident in both tracking accuracy and complexity.

8 Conclusion

In this paper we compare the computational advantages and disadvantages of two popular distributional representations for the identity management problem. We show that the two approaches are complementary - the Fourier representation is closed under prediction operations and is thus better suited for handling problems with high uncertainty while the information form representation is closed under conditioning operations and is better suited for handling problems in which a lot of observations are available. As our experiments show, using a combination of both approaches seems to often be the best approach. While converting between the two representations is a #P-hard problem in general, we show that with some of the modern permanent approximation algorithms, conversion is tractable and yields surprisingly good performance in practice.

We have focused primarily on the first-order versions of both the Fourier and information form approximations. It would be interesting to develop high order analysis with the hybrid approach.

Acknowledgement. The authors would like to thank Prof. Yinyu Ye for the helpful discussion. Thanks also to Kyle Heath for providing experimental data. Leonidas Guibas and Xiaoye Jiang wish to acknowledge the support of ARO grants W911NF-10-1-0037 and W911NF-07-2-0027, as well as NSF grant CCF 1011228 and a gift from the Google Corporation. Leonidas Guibas and Jonathan Huang acknowledge the support of grant ONR MURI N000140710747.

References

1. Agrawal, S., Wang, Z., Ye, Y.: Parimutuel betting on permutations. In: Papadimitriou, C., Zhang, S. (eds.) WINE 2008. LNCS, vol. 5385, pp. 126–137. Springer, Heidelberg (2008)
2. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 89–98 (2003)

3. Diaconis, P.: Group Representations in Probability and Statistics. Institute of Mathematical Statistics (1988)
4. Heath, K., Guibas, L.J.: Multi-person tracking from sparse 3d trajectories in a camera sensor network. In: Proceedings of IEEE ICDSC (2008)
5. Huang, B., Jebara, T.: Approximating the permanent with belief propagation. Computing Research Repository (2009)
6. Huang, J., Guestrin, C., Guibas, L.J.: Fourier theoretic probabilistic inference over permutations. Journal of Machine Learning Reserach (JMLR) 10, 997–1070 (2009)
7. Huang, J., Guestrin, C., Jiang, X., Guibas, L.J.: Exploiting probabilistic independence for permutations. In: Proceedings of the International Conference on Artificial Intelligence and Statistics, AISTATS (2009)
8. Huber, M., Law, J.: Fast approximation of the permanent for very dense problems. In: Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 681–689 (2008)
9. Jerrum, M., Sinclair, A., Vigoda, E.: A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. In: ACM Symposium on Theory of Computing, pp. 712–721 (2001)
10. Kasteleyn, P.W.: The statistics of dimers on a lattice. i. the number of dimer arrangements on a quadratic lattice. Physica, pp. 1209–1225 (1961)
11. Kondor, R., Howard, A., Jebara, T.: Multi-object tracking with representations of the symmetric group. In: Proceedings of the International Conference on Artificial Intelligence and Statistics, AISTATS (2007)
12. Ryser, H.: Combinatorial Mathematics - The Carus Mathematical Monographs Series. The Mathematical Association of America (1963)
13. Sagan, B.: The Symmetric Group: Representations, Combinatorial Algorithms, and Symmetric Functions. Springer, Heidelberg (2001)
14. Schumitsch, B., Thrun, S., Bradski, G., Olukotun, K.: The Information-Form Data Association Filter. In: Proceedings of the Neural Information Processing Systems (NIPS). MIT Press, Cambridge (2005)
15. Serre, J.-P.: Linear Representation of Finite Groups. Springer, Heidelberg (1977)
16. Shin, J., Guibas, L.J., Zhao, F.: A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks. In: Zhao, F., Guibas, L.J. (eds.) IPSN 2003. LNCS, vol. 2634, pp. 223–238. Springer, Heidelberg (2003)
17. Shin, J., Lee, N., Thrun, S., Guibas, L.J.: Lazy inference on object identities in wireless sensor networks. In: Proceeings of the International Conference on Information Processing in Sensor Networks, IPSN (2005)
18. Vontobel, P.: The bethe permanent of a non-negative matrix. In: Proceedings of the Allerton Conference on Communications, Control, and Computing (2010)
19. Watanabe, Y., Chertkov, M.: Belief propagation and loop calculus for the permanent of a non-negative matrix. Journal of Physics A: Mathematical and Theoretical 43(24), 242002 (2010)
20. Zha, H., He, X., Ding, C., Simon, H., Gu, M.: Bipartite graph partitioning and data clustering. In: Proceedings of the International Conference on Information and Knowledge Management (CIKM), pp. 25–32 (2001)

Eigenvector Sensitive Feature Selection for Spectral Clustering

Yi Jiang and Jiangtao Ren

Sun Yat-sen University, Guangzhou, 510006, P.R. China
jiangyi5@student.sysu.edu.cn, issrjt@mail.sysu.edu.cn

Abstract. Spectral clustering is one of the most popular methods for data clustering, and its performance is determined by the quality of the eigenvectors of the related graph Laplacian. Generally, graph Laplacian is constructed using the full features, which will degrade the quality of the related eigenvectors when there are a large number of noisy or irrelevant features in datasets. To solve this problem, we propose a novel unsupervised feature selection method inspired by perturbation analysis theory, which discusses the relationship between the perturbation of the eigenvectors of a matrix and its elements' perturbation. We evaluate the importance of each feature based on the average $L1$ norm of the perturbation of the first k eigenvectors of graph Laplacian corresponding to the k smallest positive eigenvalues, with respect to the feature's perturbation. Extensive experiments on several high-dimensional multi-class datasets demonstrate the good performance of our method compared with some state-of-the-art unsupervised feature selection methods.

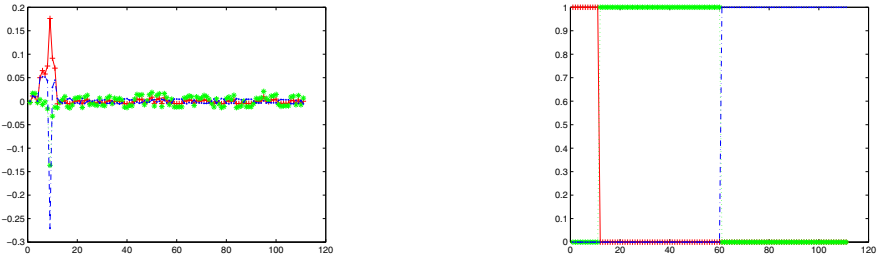
Keywords: Feature Selection, Graph Laplacian, Perturbation Analysis.

1 Introduction

Spectral clustering has wide applications ranging from text, image, web, bioinformatics to social science, for exploratory data analysis. Roughly speaking, spectral clustering is the technique to partition the rows of a matrix into multiple clusters based on the few top eigenvectors of graph Laplacian[9]. Compared with classical methods like k-means and mixture models, it has three advantages. Firstly, it doesn't need any explicit or implicit assumptions about the sample distribution. Secondly, it is easy to implement and has polynomial time solutions. Lastly, it is equivalent to graph cut problems, which are well developed. Due to these virtues, there are enormous literatures in the past on spectral clustering[6]-[21], but the nature of spectral clustering remains unchanged: *The performance of spectral clustering is determined by the quality of the chosen eigenvectors of graph Laplacian.*

However, recently, Tao Xiang, etc[10] pointed out that the first k eigenvectors of graph Laplacian may be uninformative and inappropriate for spectral clustering given noisy, irrelevant and high-dimensional data. Note that '*the first k eigenvectors*' denotes the eigenvectors corresponding to the k smallest positive eigenvalues, and '*the first k eigenvector*' denotes the eigenvector with the

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 114–129, 2011.



(a) Soft cluster indicators (b) Ideal cluster indicators

Fig. 1. The distribution of the soft and ideal cluster indicators for CLL-SUB-111

k smallest positive eigenvalue[9]. For the demonstration of the impact of irrelevant features on graph Laplacian's eigenvectors, we provide an intuitive example with a dataset *CLL-SUB-111*[1] which has 3 classes and 11340 features. In Fig.1, each curve in (a) represents the distribution of the components of one of the first three eigenvectors of its graph Laplacian computed with its full 11340 features, and the curve of the same color in (b) is the 'ideal' distribution. It is clear that each distribution in (a) has only one peak region between 0 and 20, suggesting that spectral clustering will group these samples into two clusters based on the inappropriate graph Laplacian, which differs from the 'true' cluster structure. This example demonstrates that the graph Laplacian constructed from the full features may degrade the performance of spectral clustering when there are a large number of irrelevant and noisy features in the high-dimensional dataset, hence we need to perform feature selection before constructing the graph Laplacian for spectral clustering.

The core problem of feature selection is how to evaluate the importance of features, which has numerous criterions such as Laplacian Score(LS)[36], Spec[37], MCFS[39], FSFS[34], FCBF[35], FSSEM[30] and EVSC[41], etc. In the recent development of spectral clustering, Ling Huang, etc[11]-[13] present some proofs of the close relationship between the perturbation of clustering result and laplacian graph's eigenvectors due to the perturbation of data. These researches inspire us that the perturbation of the feature values of data will have impact on the perturbation of the eigenvectors of graph Laplacian and the result of spectral clustering, hence we can evaluate the importance of features by using the perturbation of the eigenvectors of graph Laplacian in respect of the perturbation of each feature.

In this paper, we propose a new feature evaluation criterion based on the recent developments of perturbation analysis[2] [11]-[15]. Specifically, to evaluate a feature's importance, we perturb the value of this feature by introducing a perturbation factor to it for all the samples in the data set. This will induce a perturbation of the similarity matrix, and in turn a perturbation of the graph Laplacian. Finally, this leads to the perturbation of the eigenvectors of graph Laplacian. It is natural to believe that if a small perturbation of one feature

[1] `http://featureselection.asu.edu/datasets.php`

induces a great perturbation of the eigenvectors of graph laplacian, this feature is important for spectral clustering. Then, we use the average L1-norm of the perturbation of the first k eigenvectors of graph Laplacian in terms of the small perturbation of one feature to estimate the significance of this feature. The criterion is referred to as *EigenVector Sensitive Feature Selection Criterion (EVSFSC)*. Based on this criterion, we can perform feature selection for spectral clustering. Extensive experiment results over six real-world datasets demonstrate the superiority of our method compared with four traditional unsupervised feature selection methods.

2 Feature Selection Based on Perturbation Analysis

In this section, we study the perturbation of graph Laplacian's eigenvectors in terms of the perturbation of each feature, with three different definitions of graph Laplacian L, L_{rw} and L_{sym} [9]. Based on these analysis, we formulate three feature evaluation criterions, then a feature selection algorithm is proposed for the most common spectral clustering algorithms.

2.1 Problem Definition

For a dataset $X = \{x^i\}_{i=1}^n$, $x^i \in R^{K\times 1}$ represents the i-th data sample, where K is the dimension of X, and x_t^i denotes the t-th feature value of x^i. Suppose S, D and L are similarity matrix, diagonal degree matrix and graph Laplacian respectively, $S_{i,j}$ represents the similarity between x^i and x^j, $D = diag(S\mathbf{1})$ ($\mathbf{1} = (1, ..., 1)^T$) and $L = D - S$.

Let ξ be a perturbation factor, if we perturb X on the t-th feature with ξ, which means $\hat{x}_t^i = x_t^i + \xi x_t^i$, $i = 1, ..., n$, and keep other features unchanged, then we get a perturbed dataset $\hat{X}_t = \{\hat{x}^i\}_{i=1}^n$. Let $\hat{L}_t$ be the perturbed graph Laplacian based on $\hat{X}_t$. Suppose $\hat{q}_{t,r}$ and q_r are the r-th eigenvector of $\hat{L}_t$ and L respectively, then the perturbation of the r-th eigenvector of graph Laplacian L caused by the perturbation of the t-th feature can be defined as $\triangle q_{t,r} = \hat{q}_{t,r} - q_r$. The greater the L1 norm of $\triangle q_{t,r}$ is, the more important the t-th feature is. Thus, our main problem is how to evaluate $\triangle q_{t,r}$ with respect to ξ. Let's begin by proving the relationship between $\hat{D}_t, \hat{L}_t$ and D, L, where $\hat{D}_t$ is the perturbed similarity matrix based on $\hat{X}_t$.

In this paper, we adopt RBF function as the similarity measure between data samples, and our framework can also be easily extended to other popular similarity measures such as dot product, square Euclidean, etc. Then $S_{i,j}$ can be formulated as

$$S_{i,j} = e^{-\frac{\sum_{h=1}^{K}(x_h^i - x_h^j)^2}{2\delta^2}}$$

where δ^2 is the kernel bandwidth. When we perturb the t-th feature with factor ξ, which means $\hat{x}_t^i = (1+\xi)x_t^i$, $i = 1, ..., n$, the perturbed similarity $\hat{S}_{t,i,j}$ is

$$\hat{S}_{t,i,j} = e^{-\frac{\sum_{h=1,h\neq t}^{K}(x_h^i - x_h^j)^2 + (1+\xi)^2(x_t^i - x_t^j)^2}{2\delta^2}} \tag{1}$$

Now we can derive the relationship between $\hat{D}_t, \hat{L}_t$ and D, L as follows.

Theorem 1. *If $\xi \to 0$, $\hat{D}_t$ and $\hat{L}_t$ can be approximated by*

$$\hat{D}_t \approx D - \xi D_t^1, \quad \hat{L}_t \approx L - \xi L_t^1$$

Then,

$$\hat{D}_t - D \approx -\xi D_t^1, \quad \hat{L}_t - L \approx -\xi L_t^1 \tag{2}$$

where $S_{t,i,j}^1 = S_{i,j}\frac{(x_t^i - x_t^j)^2}{\delta^2}$, $D_{t,i,i}^1 = \sum_{h=1}^n S_{t,i,h}^1$, $L_t^1 = D_t^1 - S_t^1$, D_t^1 is a diagonal matrix.

Proof. based on formula (1), we can get

$$\frac{\partial \hat{S}_{t,i,j}}{\partial \xi} = -(\xi + 1)\hat{S}_{t,i,j}\frac{(x_t^i - x_t^j)^2}{\delta^2}$$

Then, when $\xi \to 0$, we can derive the first-order Taylor expansion for $\hat{S}_{t,i,j}$

$$\hat{S}_{t,i,j} = S_{i,j} - S_{i,j}\frac{(x_t^i - x_t^j)^2}{\delta^2} \cdot \xi + O(\xi)$$

and we can get

$$\hat{S}_{t,i,j} - S_{i,j} \approx -S_{i,j}\frac{(x_t^i - x_t^j)^2}{\delta^2} \cdot \xi \approx -\xi \cdot S_{t,i,j}^1$$

$$\hat{D}_{t,i,i} = \sum_{h=1}^n \hat{S}_{t,i,h} \approx \sum_{h=1}^n S_{i,h} - \xi \cdot \sum_{h=1}^n S_{t,i,h}^1 \approx D_{i,i} - \xi \cdot D_{t,i,i}^1$$

Thus,

$$\hat{D}_t - D \approx -\xi D_t^1$$
$$\hat{L}_t - L = (\hat{D}_t - D) - (\hat{S}_t - S) = -\xi \cdot (\hat{D}_t^1 - \hat{S}_t^1) \approx -\xi L_t^1$$

□

In general, $L = D - S$ is the unnormalized graph Laplacian [9]. Moreover, there are two other normalized graph Laplacians [9] $L_{rw} = D^{-1}L$ and $L_{sym} = D^{-1/2}LD^{-1/2}$. We will derive $\triangle q_{t,r}$, $\triangle q_{rw,t,r}$ and $\triangle q_{sym,t,r}$ with respect to L, L_{rw} and L_{sym} respectively in the following sections.

2.2 $\triangle q_{t,r}$ with Respect to L

Perturbation analysis theory [2] discusses the relationship between the perturbation of the eigenvectors of a matrix and its elements' perturbation, which will be summarized in **Theorem 2**.

Theorem 2. *(*First-Order Eigenvector Perturbation*)*
Let A and B be matrices with elements which satisfy the relations: $|A_{ij}| < 1$ and $|B_{ij}| < 1$, and A has the normalized eigenvector set $\{q_r\}_{r=1}^n$ and eigenvalue set $\{\lambda_r\}_{r=1}^n$, where the multiplicity of any eigenvalue is 1, if $\xi \to 0$, then the r-th eigenvector $\hat{q}_r$ of $A + \xi B$ is approximately expressed as:

$$\hat{q}_r \approx q_r + \xi \cdot \{\sum_{h=1, h\neq r}^n \frac{q_h^T B q_r}{\lambda_r - \lambda_h} q_h\}. \tag{3}$$

Based on **Theorem 1** and **Theorem 2**, we can easily derive $\triangle q_{t,r} = \hat{q}_{t,r} - q_r$, which is summarized in **Theorem 3**. It is worth pointing out that the conditions $|A_{ij}| < 1$ and $|B_{ij}| < 1$ can be satisfied for RBF kernel.

Theorem 3. *Let* $\{\lambda_r\}_{r=1}^n$ *and* $\{q_r\}_{r=1}^n$ *be the eigenvalue and normalized eigenvector sets of* $Lq_r = \lambda_r q_r$, *and* $\lambda_1 < \lambda_2 < \ldots < \lambda_n$, *if* $\xi \to 0$, *then the r-th eigenvector of* $\hat{L}_t$ *based on* $\hat{X}_t$ *can be approximated by*

$$\hat{q}_{t,r} \approx q_r + \xi \cdot p_{t,r}$$

Then,

$$\triangle q_{t,r} = \hat{q}_{t,r} - q_r \approx \xi \cdot p_{t,r} \tag{4}$$

where

$$p_{t,r} = -\sum_{h=1, h\neq r}^{n} \Big(\frac{q_h^T L_t^1 q_r}{\lambda_r - \lambda_h} \Big) q_h$$

Proof. this can be proved with **Theorem 1** and **Theorem 2**. □

2.3 $\triangle q_{rw,t,r}$ with Respect to L_{rw}

For computing the r-th eigenvector's perturbation $\triangle q_{rw,t,r} = \hat{q}_{rw,t,r} - q_{rw,r}$ of L_{rw}, where $q_{rw,r}$ is the r-th eigenvector of L_{rw} based on X, and $\hat{q}_{rw,t,r}$ is the r-th eigenvector of $\hat{L}_{rw,t}$ based on $\hat{X}_t$, we first borrow the following definition from [4], which provides some solutions for the algebraic eigenvalue problems.

Definition 1 Hermitian Definite Pencil[4]
A Hermitian definite pencil{A,B} ($A \in R^{n\times n}$ and $A \in R^{n\times n}$) is a generalized Hermitian eigenvalue problem: $Aq = \lambda Bq$, where A and B are Hermitian, that is, **if the conjugate transpose of matrix A or B is denoted by** A^* **or** B^*, then $A^* = A$ and $B^* = B$, and A or B or $\alpha A + \beta B$ for some scalars α and β is positive definite, q and λ are the corresponding eigenvector and eigenvalue respectively.

Since $L = L^*$ and $\forall$ x, $x^T Dx > 0$, then $\{L, D\}$ is a Hermitian definite pencil. For the proof of **Theorem 4**, we describe one property for L_{rw} and two properties for $\{L, D\}$ in **Property 1**, which can be found in [9] and [3] respectively.

Property 1
(a) The eigen-system of L_{rw} is equal to that of the Hermitian definite pencil $\{L, D\}$. That is, $\forall\ r \in \{1, ..., n\}$, $L_{rw} q_{rw,r} = \lambda_{rw,r} q_{rw,r} \Leftrightarrow Lq_{rw,r} = \lambda_{rw,r} D q_{rw,r}$

(b) For $\{L, D\}$, if $\lambda_{rw,r} \neq \lambda_{rw,r1}$, $q_{rw,r}^T D q_{rw,r1} = 0$, and if $q_{rw,r}$ is a normalized eigenvector, then $q_{rw,r}^T D q_{rw,r} = 1$.

(c) If $\{L, D\}$ has the eigenvalue and eigenvector sets $\{\lambda_{rw,r}\}_{r=1}^n$ and $\{q_{rw,r}\}_{r=1}^n$, and the multiplicity of any eigenvalue is 1, then $\{q_{rw,r}\}_{r=1}^n$ constitute a basis for R^n.

Then we can propose **Theorem 4** for $\triangle q_{rw,t,r}$ in the following.

Theorem 4. *For* $\{L,D\}$, *let* $\{\lambda_{rw,r}\}_{r=1}^n$ *and* $\{q_{rw,r}\}_{r=1}^n$ *be the corresponding eigenvalue and normalized eigenvector sets, and* $\lambda_{rw,1} < \lambda_{rw,2} < \ldots < \lambda_{rw,n}$, *if*

$\xi \to 0$, the r-th perturbed eigenvector $\hat{q}_{rw,t,r}$ of the perturbed normalized graph Laplacian $\hat{L}_{rw,t}$ based on $\hat{X}_t$ can be approximated as

$$\hat{q}_{rw,t,r} = q_{rw,r} + \xi \cdot p_{rw,t,r} + O(\xi \cdot \mathbf{1})$$

Then,

$$\triangle \mathbf{q_{rw,t,r}} = \hat{\mathbf{q}}_{\mathbf{rw,t,r}} - \mathbf{q_{rw,r}} = \xi \cdot \mathbf{p_{rw,t,r}} + \mathbf{O}(\xi \cdot \mathbf{1}) \tag{5}$$

where $p_{rw,t,r} = \left\{ \sum_{h=1, h\neq r}^{n} \left(\frac{q_{rw,h}^T \{\lambda_{rw,r} D_t^1 - L_t^1\} q_{rw,r}}{\lambda_{rw,r} - \lambda_{rw,h}} \right) q_{rw,h} + \left(\frac{q_{rw,r}^T D_t^1 q_{rw,r}}{2} \right) q_{rw,r} \right\}$.

Proof. Based on **Theorem 1**, if $\xi \to 0$, then

$$\hat{D}_t - D = -\xi \cdot D_t^1 + O(\xi \cdot \mathbf{I}) \quad and \quad \hat{L}_t - L = -\xi \cdot L_t^1 + O\{\xi \cdot (\mathbf{1}^T \cdot \mathbf{1})\}.$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{1} = (1, ..., 1)^T$.
It is natural that[2]

$$\hat{\lambda}_{rw,t,r} - \lambda_{rw,r} = \xi \cdot \eta_{rw,t,r} + O(\xi \cdot \mathbf{1}) \tag{6}$$

$$\hat{q}_{rw,t,r} - q_{rw,r} = \xi \cdot p_{rw,t,r} + O(\xi \cdot \mathbf{1}) \tag{7}$$

Now our goal is to estimate the column vector $p_{rw,t,r}$.
Because of **Property 1 (a)**, we get

$$\hat{L}_t \hat{q}_{rw,t} = \hat{\lambda}_{rw,t} \hat{D}_t \hat{q}_{rw,t} \tag{8}$$

Based on formula (2) and (6)-(8), we get

$$\{L - \xi \cdot L_t^1\}\{q_{rw,r} + \xi \cdot p_{rw,t,r}\} = \{\lambda_{rw,r} + \xi \cdot \eta_{rw,t,r}\}\{D - \xi \cdot D_t^1\}\{q_{rw,r} + \xi \cdot p_{rw,t,r}\} \tag{9}$$

When $\xi \to 0$, (9) can be rewritten as

$$L p_{rw,t,r} - L_t^1 q_{rw,r} = -\lambda_{rw,r} D_t^1 q_{rw,r} + \lambda_{rw,r} D p_{rw,t,r} + \eta_{rw,t,r} D q_{rw,r} \tag{10}$$

With **Property 1**(c), $p_{rw,t,r}$ can be expressed as a linear combination of $\{q_{rw,r}\}_{r=1}^n$, that is,

$$p_{rw,t,r} = \sum_{h=1}^{n} \varepsilon_{r,h} q_{rw,h} \tag{11}$$

Substitute (11) into (10), and left multiply (10) by $q_{rw,r1}^T (r1 \neq r)$, we get

$$\begin{aligned} \sum_{h=1}^{n} \lambda_{rw,h} \varepsilon_{r,h} q_{rw,r1}^T D q_{rw,h} - q_{rw,r1}^T L_t^1 q_{rw,r} = -\lambda_{rw,r} q_{rw,r1}^T D_t^1 q_{rw,r} + \\ \lambda_{rw,r} \sum_{h=1}^{n} \varepsilon_{r,h} q_{rw,r1}^T D q_{rw,h} + \eta_{rw,t,r} q_{rw,r1}^T D q_{rw,r} \end{aligned} \tag{12}$$

With **Property 1** (b), we get

$$\varepsilon_{r,r1} = \frac{q_{rw,r1}^T \{\lambda_{rw,r} D_t^1 - L_t^1\} q_{rw,r}}{\lambda_{rw,r} - \lambda_{rw,r1}} \tag{13}$$

For $\{\hat{L}_t, \hat{D}_t\}$, which is also a Hermitian definite pencil,

$$\{q_{rw,r}^T + \xi p_{rw,t,r}^T\}\{D - \xi D_t^1\}\{q_{rw,r} + \xi p_{rw,t,r}\} = 1 \tag{14}$$

With $\xi \to 0$ and (11), (14) can be rewritten as

$$\varepsilon_{rr} = \frac{q_{rw,r}^T D_t^1 q_{rw,r}}{2} \tag{15}$$

Finally, based on (13) and (15), $p_{rw,t,r}$ can be computed by

$$p_{rw,t,r} = \Big\{ \sum_{h=1,h\neq r}^{n} \Big(\frac{q_{rw,h}^T \{\lambda_{rw,r} D_t^1 - L_t^1\} q_{rw,r}}{\lambda_{rw,r} - \lambda_{rw,h}} \Big) q_{rw,h} + \Big(\frac{q_{rw,r}^T D_t^1 q_{rw,r}}{2} \Big) q_{rw,r} \Big\}$$

□

2.4 $\triangle q_{sym,t,r}$ with Respect to L_{sym}

The spectral clustering theories [9] reveal that if $q_{rw,r}$ is the eigenvector of L_{rw} with $\lambda_{rw,r}$, then $q_{sym,r} = D^{1/2} q_{rw,r}$ is the eigenvector of L_{sym} with the same eigenvalue. Based on this connection between the eigen-system of L_{rw} and L_{sym}, and **Theorem 4**, the calculation of $\triangle q_{sym,t,r}$ for L_{sym} is shown in **Theorem 5**.

Theorem 5. *With the conditions of **Theorem 4**, let $q_{sym,r}$ be the normalized eigenvector of L_{sym}, and $\hat{q}_{sym,t,r}$ be the corresponding r-th eigenvector of $\hat{L}_{sym,t}$ based on $\hat{X}_t$, if $\xi \to 0$, then $\hat{q}_{sym,t,r}$ can be approximated as*

$$\hat{q}_{sym,t,r} = q_{sym,r} + \xi \cdot p_{sym,t,r} + O(\xi \cdot \mathbf{1})$$

Then,

$$\triangle \mathbf{q_{sym,t,r}} = \hat{\mathbf{q}}_{\mathbf{sym,t,r}} - \mathbf{q_{sym,r}} = \xi \cdot \mathbf{p_{sym,t,r}} + \mathbf{O}(\xi \cdot \mathbf{1}) \tag{16}$$

where $p_{sym,t,r} = \Big\{ -\frac{1}{2} D^{-1/2} D_t^1 q_{rw,r} + D^{1/2} p_{rw,t,r} \Big\}$

Proof. If $\xi \to 0$, then

$$\hat{D}_t^{1/2} = \{D - \xi \cdot D_t^1 + O(\xi \cdot \mathbf{I})\}^{1/2} \approx D^{1/2} (\mathbf{I} - \xi \cdot D^{-1} D_t^1)^{1/2} \tag{17}$$

where $\mathbf{I}$ is the identity matrix.
Then, the first order Taylor expansion of $\hat{D}_t^{1/2}$ can be rewritten as

$$(\mathbf{I} - \xi \cdot D^{-1} D_t^1)^{1/2} \approx \mathbf{I} - \frac{\xi}{2} \cdot D^{-1} D_t^1 + O(\xi \cdot \mathbf{I}) \tag{18}$$

Based on formula (17) and (18), we get

$$\hat{D}_t^{1/2} = D^{1/2} - \frac{\xi}{2} \cdot D^{-1/2} D_t^1 + O(\xi \cdot \mathbf{I})$$

Thus

$$\hat{q}_{sym,t,r} = \hat{D}_t^{1/2} \hat{q}_{rw,t,r} = q_{sym,r} + \xi \cdot \Big\{ -\frac{1}{2} D^{-1/2} D_t^1 q_{rw,r} + D^{1/2} p_{rw,t,r} \Big\} + O(\xi \cdot \mathbf{1})$$

□

2.5 Eigenvector Sensitive Feature Selection

Based on the discussion of section 2.2, 2.3 and 2.4, when the value of ξ is sufficiently small, the j-th component of the eigenvector perturbation $\triangle q_{t,r}$($\triangle q_{rw,t,r}$ or $\triangle q_{sym,t,r}$) of graph Laplacian L(L_{rw} or L_{sym}) is approximately linear with ξ, and the corresponding gradient is just the j-th component of $p_{t,r}$($p_{rw,t,r}$ or $p_{sym,t,r}$), which reflects the rate of the change of the j-th component of the r-th eigenvector of L(L_{rw} or L_{sym}) in response to the perturbation of the t-th feature. Hence, it is natural to use the $L1$ norm of $p_{t,r}$, $p_{rw,t,r}$ and $p_{sym,t,r}$ to evaluate the importance of the t-th feature to the r-th eigenvector of L, L_{rw} and L_{sym} respectively.

However, since the result of spectral clustering is determined by the first k eigenvectors of graph Laplacian, we should evaluate the importance of the r-th feature to the spectral clustering based on its impact on the first k eigenvectors of the corresponding graph Laplacian. Thus, we propose to employ the average $L1$ norm of $p_{t,r}$($p_{rw,t,r}$ or $p_{sym,t,r}$) over the first k eigenvectors of L(L_{rw} or L_{sym}) to estimate the importance of the t-th feature in the corresponding spectral clustering. This criterion is called *EigenVector Sensitive Feature Selection Criterion*(***EVSFSC***), whose formal definitions are expressed as follows.

When the graph Laplacian is L, for the t-th feature, then

$$EVSFSC(t) = \frac{1}{k} \sum_{r=2}^{k+1} \|p_{t,r}\|_1 \tag{19}$$

Similarly, when the graph Laplacian is L_{rw}, for the t-th feature, then

$$EVSFSC(t) = \frac{1}{k} \sum_{r=2}^{k+1} \|p_{rw,t,r}\|_1 \tag{20}$$

Finally, when the graph Laplacian is L_{sym}, for the t-th feature, then

$$EVSFSC(t) = \frac{1}{k} \sum_{r=2}^{k+1} \Big\| p_{sym,t,r} \|_1 \tag{21}$$

Algorithm 1. Eigenvector Sensitive Feature Selection for Spectral Clustering

Input: Data set **X**, Feature number **m**, Spectral clustering type **SCT**
Output: Feature subset F_m
1. Construct the similarity matrix S with RBF function
2. Build L and D based on S
3. If **SCT** $==$' **USC**'
4. Calculate the eigen-system (λ_r, q_r) of L, $1 \leq r \leq n$.
5. Else if **SCT**=='**NSCLrm**'
6. Calculate the eigen-system (λ_r, q_r) of $L_{rw} = D^{-1}L$, $1 \leq r \leq n$,
7. Else
8. Calculate the eigen-system (λ_r, q_r) of $L_{sym} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}$, $1 \leq r \leq n$.
9. End if
10. Normalize the eigenvectors of $\{q_r\}_{r=1}^{n}$.
for $t = 1$ **to** K **do**
11. Calculate the EVSFSC of the t-th feature based on (19), (20) or (21).
end for
12. Rank the features decreasingly according to the value of EVSFSC and select the leading m features, that is $F_m = \{F_{K_1}, ..., F_{K_m}\}$
13. return F_m

2.6 Eigenvector Sensitive Feature Selection for Spectral Clustering

Based on the criterions of (19)-(21), we summarize the eigenvector sensitive feature selection for spectral clustering algorithm in **Algorithm 1**. In this algorithm, **SCT** represents the type of spectral clustering, **USP** represents the unnormalized spectral clustering, **NSCLrm** and **NSCLsym** represent the normalized spectral clustering with L_{rm} and L_{sym} respectively. The computation complexity for main steps is listed below.

• In step 1 and 2, we need $O(n^2K)$ operations to build S, D and L;
• In step 4, 6 or 8, we need $O(n^3)$ operations to get the eigenvalues and eigenvectors of graph Laplacian by Lanczos algorithm[5];
• In step 10, we need $O(n^3K)$ operations to calculate the EVSFSC score for all features;
• In step 11, the top m features can be found within O(KlogK).
Thus, the computation complexity of **Algorithm 1** is $MAX(n^3K, KlogK)$.

3 Related Work

Spectral Clustering. The spectral clustering based on the graph cut theory is to find the best cuts of a graph according to certain predefined criterion functions such as RatioCut[6] and normalized cut(Ncut)[7]. The relaxing RatioCut leads to the unnormalized spectral clustering[9] based on the eigenvectors of unnormalized graph Laplacian $L = D - S$, while the relaxing Ncut leads to the normalized spectral clustering[7][8] based on the eigenvectors of $L_{rw} = D^{-1}L$ or $L_{sym} = D^{-1/2}LD^{-1/2}$. Recently, there are several works focusing on the impact

of small errors in data or similarity matrix on spectral clustering, based on ***the perturbation analysis*** [2]. [11]-[13] derive some approximate upper bounds on the errors of k-way spectral clustering with respect to the small change of data or similarity matrix($k = 2, 3, ...$). Another line of this works is to update the information of the eigen-system of graph Laplacian in the incremental spectral clustering[14] [15], given a small change of similarity matrix. Besides, there are enormous literatures discussing other subjects like the incorporation of user supervision into spectral clustering[16]-[18], and the strategy of constructing graph Laplacian for spectral clustering[19]-[21], etc.

Unsupervised Feature Selection. Most of existing methods can be classified into the three categories. Methods in the first category are wrapper approaches. These include unsupervised feature selections for K-means[22]-[25], Mixture Models[26]-[32] and PCA(Principal Components Analysis)[33]. The second category measures feature similarity based some criterions, whereby redundant features are removed. [34] and [35] are the two representatives of this kind. The third category is the spectral methods. [36]-[38] perform feature selection based on certain evaluation criterions, which are the function of the eigen-system of graph Laplacian. More recently, in [39] and [40], the feature selection problems are transformed into the regression problems, which aim to find those feature vectors aligning closely to the few top eigenvectors of graph Laplacian. In our previous work [41], a eigenvalue sensitive feature selection method is proposed. But it is different from the method of this paper. The core idea of [41] is that the feature importance should be evaluated by the gradient of the eigenvalue of graph Laplacian with respect to the weight of feature. But in this paper, we introduce the perturbation analysis theory.

4 Empirical Analysis

In this section, we perform extensive experiments to demonstrate the performance of our proposed feature selection method comparing to several popular unsupervised feature selection methods. They are FSFS[34], Laplacian Score(LS)[36], Spec[37] and MCFS[39].

4.1 Dataset Decription

Six high-dimensional and multi-class datasets are selected for the experiments, which are briefly described in Table 1. All of the datasets can be found from the Feature Selection Repository[2]. For simpleness, we use **CLL**, **ORL**, **PIX**, **TOX**, **AR** and **PIE** to represent the data sets CLL-SUB-111, orlraws10P, pixraw10P, TOX-171, warpAR10P and warpPIE10P respectively.

4.2 Evaluation Criterion

In the experiments, **Clustering Accuracy(CA)**[36] is used to evaluate the performance of spectral clustering. Based on the comparison between the predefined

[2] `http://featureselection.asu.edu/datasets.php`

Table 1. Summary of six datasets

Data Set	*Instances*	*Features*	*Classes*
CLL-SUB-111	111	11340	3
orlraws10P	100	10304	10
pixraw10P	100	10000	10
TOX-171	171	5748	4
warpAR10P	130	2400	10
warpPIE10P	210	2420	10

labels **c(i)** of all samples and the obtained labels **sc(i)** by spectral clustering, **Clustering Accuracy(CA)** is formally defined as

$$\mathbf{CA} = \frac{\sum_{i=1}^{n} \delta(\mathbf{c(i)}, map(\mathbf{sc(i)}))}{n}$$

where n is the total number of data points and $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $map(\mathbf{sc(i)})$ is the permutation mapping function that maps each cluster label **sc(i)** to the equivalent label from data corpus. Here, we use the Kuhn-Munkres algorithm[1] as the mapping function.

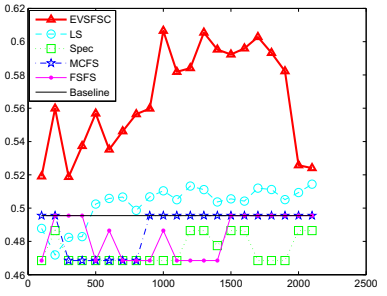
4.3 Experiment Setup

Four popular unsupervised feature selection methods are chosen as baseline methods, which are FSFS[34], Laplacian Score(LS)[36], Spec[37] and MCFS[39], and their matlab codes can be found at their homepages[3]. As discussed above, we choose **RBF function** as similarity measure, whose parameter is determined by cross-validation. Then for each dataset, the four baseline criterions and **EVSFSC** are used to select the best 100, 200,...,2100 features. Based on the selected feature subsets, the **Clustering Accuracy** of unnormalized spectral clustering with L and normalized spectral clustering with L_{rw} and L_{sym} are demonstrated in Fig.2, Fig.3 and Fig.4 respectively. And as a baseline, the **Clustering Accuracy** with the full features (without feature selection) is also depicted in all the figures, and it is referred to as 'Baseline' in the figures.

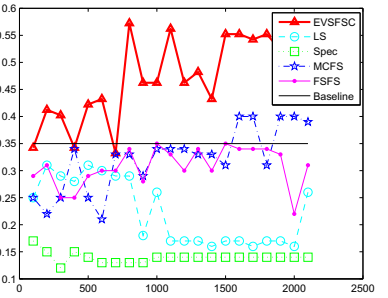
4.4 Experiment Results

Unnormalized spectral clustering with L Fig. 2(a-f) show the curves of the **Clustering Accuracy** of unnormalized spectral clustering with L versus the number of selected features on six datasets respectively, based on FSFS, Laplacian Score(LS), Spec, MCFS and EVSFSC. As we can see, our proposed algorithm achieves consistently better performance than the other methods and

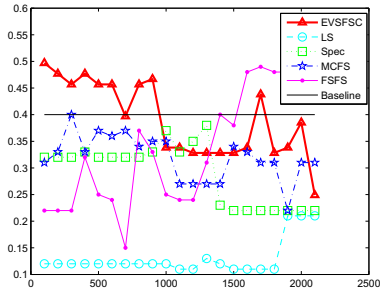
[3] `http://www.facweb.iitkgp.ernet.in/~pabitra/paper.html`, `http://www.zjucadcg.cn/dengcai/MCFS/index.html`, `http://featureselection.asu.edu/software.php`



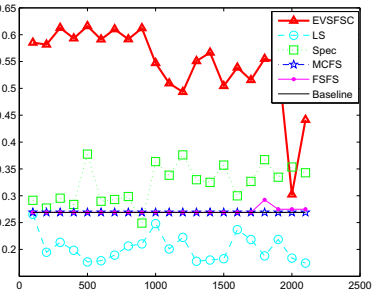
(a) CLL-SUB-111



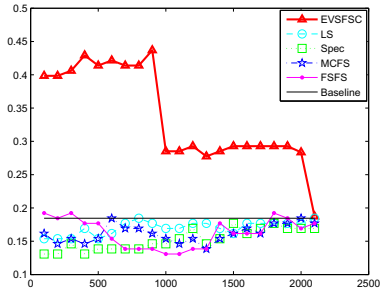
(b) Orlraws10P



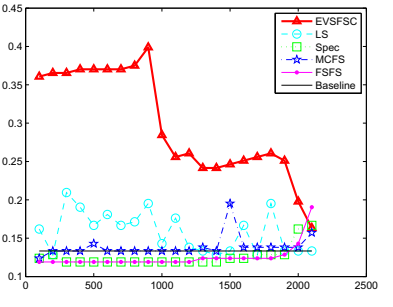
(c) Pixraw10P



(d) TOX-171



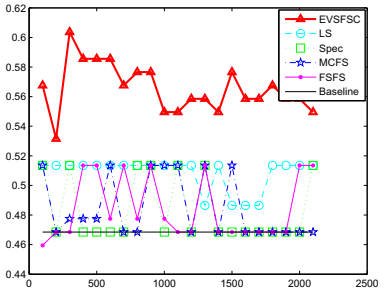
(e) WarpAR10P



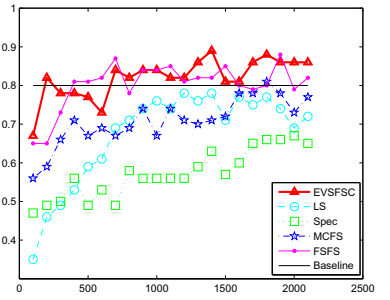
(f) WarpPIE10P

Fig. 2. Unnormalized Spectral Clustering with L

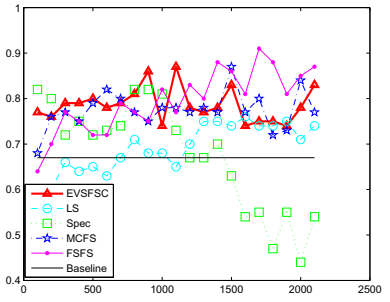
the baseline method without feature selection. Although the unnormalized spectral clustering with all features produces a poor result, most of the existing feature selection methods don't produce much better results, and sometimes produce even worse results, for example in Figure 2(b), (c) and (e). However, our method can use less than 1000 features to produce reasonably good results, whose **Clustering Accuracy** is generally higher than 0.6 on **CLL**, **ORL** and **TOX** datasets. Especially for **PIX**, **AR** and **PIE** datasets, only several hundred of selected features by our method can achieve the best results, compared with other methods.



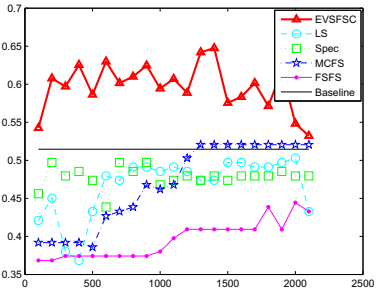
(a) CLL-SUB-111



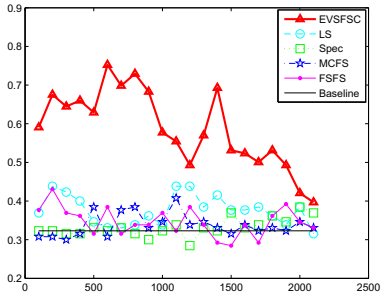
(b) Orlraw10P



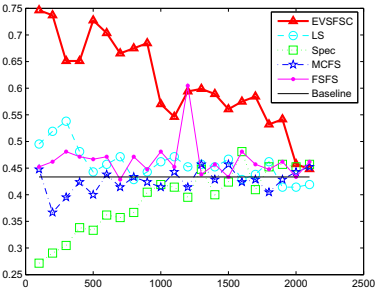
(c) Pixraw10P



(d) TOX-171



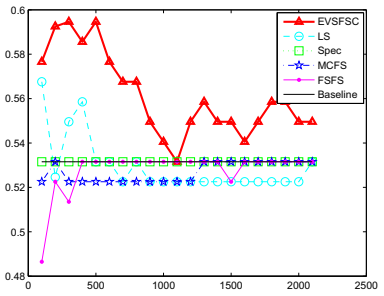
(e) WarpAR10P



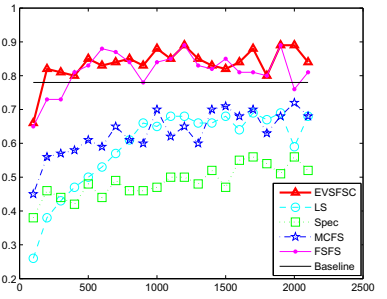
(f) WarpPIE10P

Fig. 3. Normalized Spectral Clustering with L_{rm}

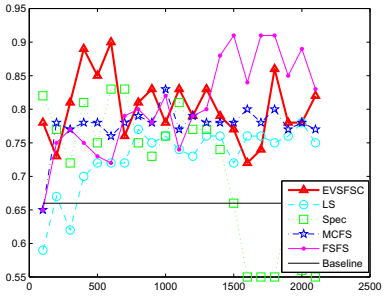
Normalized spectral clustering with L_{rw} Fig. 3(a-f) reveal the curves of the **Clustering Accuracy** of normalized spectral clustering with L_{rw} versus the number of selected features on six data sets respectively, based on **EVSFSC** and other four methods. For all of the six data sets, our method also can achieve best performance than the others. Specifically, the difference between 'Baseline' and FSFS, Laplacian Score(LS), Spec, MCFS is not obvious on **CLL**, **TOX**, **AR** and **PIE** datasets, but **EVSFSC** can still achieve great improvements.



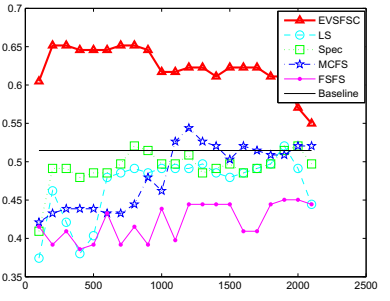
(a) CLL-SUB-111



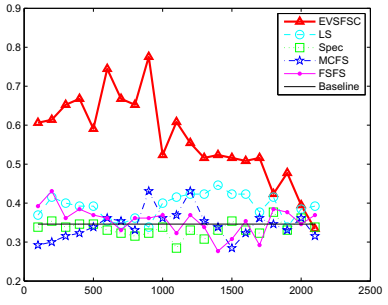
(b) Orlraws10P



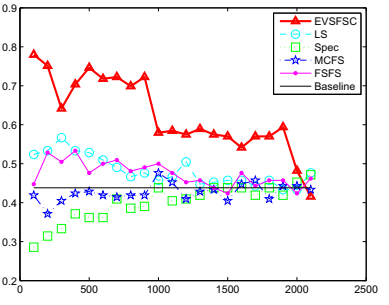
(c) Pixraw10P



(d) TOX-171



(e) WarpAR10P



(f) WarpPIE10P

Fig. 4. Normalized Spectral Clustering with L_{sym}

Normalized spectral clustering with L_{sym} Fig. 4(a-f) demonstrate the curves of the **Clustering Accuracy** of Normalized spectral clustering algorithm with L_{sym} versus the number of selected features on six data sets respectively, based on **EVSFSC** and other four methods mentioned before. Except for datasets **ORL** and **PIX**, our method significantly outperforms the other four methods. On data sets **ORL** and **PIX**, there exist some methods such as FSFS and MCFS performing comparably to our method with the increase of feature number, but EVSFSC can achieve the same good results with fewer features than them.

5 Conclusion

In this paper, we propose a new feature selection criterion, called *EVSFSC*, for spectral clustering. *EVSFSC* evaluates the importance of each feature by its impact on the eigenvectors of graph Laplacian with perturbation analysis theory. The extensive experiments demonstrate the excellent performance of our method, compared with four state-of-the-art methods.

Acknowledgements. This work was supported by National Natural Science Foundation of China under Grant No. 60703110.

References

1. Lovasz, L., Plummer, M.: Matching Theory (1986)
2. Wilkinson, J.H.: The Algebraic Eigenvalue Problem Numerical Mathematics and Scientific Computation, Oxford, pp. 62–104 (1988)
3. Joel, N.: Franklin: Matrix Theory (2000)
4. Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., van der Vorst, H. (eds.): Templates for the solution of Algebraic Eigenvalue Problems: A Practical Guide. SIAM, Philadelphia (2000)
5. Stewart, G.W.: Matrix Algorithms Volumn II: Eigensystems. SIAM, Philadelphia (2001)
6. Hagen, L.W., Kahng, A.B.: New spectral methods for ratio cut partitioning and clustering. IEEE Trans. on CAD of Integrated Circuits and Systems (TCAD) 11(9), 1074–1085 (1992)
7. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell (PAMI) 22(8), 888–905 (2000)
8. Ng, A.Y., Jordan, M.I., Weiss, Y.: On Spectral Clustering: Analysis and an algorithm. In: NIPS 2001, pp. 849–856 (2001)
9. von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing (SAC) 17(4), 395–416 (2007)
10. Xiang, T., Gong, S.: Spectral clustering with eigenvector selection. Pattern Recognition (PR) 41(3), 1012–1029 (2008)
11. Huang, L., Yan, D., Jordan, M.I., Taft, N.: Spectral Clustering with Perturbed Data. In: NIPS 2008, pp. 705–712 (2008)
12. Yan, D., Huang, L., Jordan, M.I.: Fast approximate spectral clustering. In: KDD 2009, pp. 907–916 (2009)
13. Hunter, B., Strohmer, T.: Performance Analysis of Spectral Clustering on Compressed, Incomplete and Inaccurate Measurements CoRR abs/1011.0997 (2010)
14. Gong, Y., Huang, T.S.: Incremental Spectral Clustering With Application to Monitoring of Evolving Blog Communities. In: SDM (2007)
15. Ning, H., Xu, W., Chi, Y., Gong, Y., Huang, T.S.: Incremental spectral clustering by efficiently updating the eigen-system. Pattern Recognition (PR) 43(1), 113–127 (2010)
16. Song, X., Zhou, D., Hino, K., Tseng, B.L.: Evolutionary spectral clustering by incorporating temporal smoothness. In: KDD 2007, pp. 153–162 (2007)
17. Coleman, T., Saunderson, J., Wirth, A.: Spectral clustering with inconsistent advice. In: ICML 2008, pp. 152–159 (2008)

18. Wang, X., Davidson, I.: Flexible constrained spectral clustering. In: KDD 2010, pp. 563–572 (2010)
19. Bach, F.R., Jordan, M.I.: Learning Spectral Clustering. In: NIPS (2003)
20. Ozertem, U., Erdogmus, D., Jenssen, R.: Mean shift spectral clustering. Pattern Recognition (PR) 41(6), 1924–1938 (2008)
21. Bhler, T., Hein, M.: Spectral clustering based on the graph p-Laplacian. In: ICML, p. 11 (2009)
22. Kim, Y., Street, W.N., Menczer, F.: Feature selection in unsupervised learning via evolutionary search. In: KDD 2000, pp. 365–369 (2000)
23. Modha, D., Spangler, S.: Feature Weighting in k-Means Clustering. Machine Learning (2002)
24. Huang, J.Z., Ng, M.K., Rong, H., Li, Z.: Automated Variable Weighting in k-Means Type Clustering. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 27(5), 657–668 (2005)
25. Boutsidis, C., Mahoney, M.W., Drineas, P.: Unsupervised Feature Selection for the K-means Clustering Problem. In: NIPS 2009 (2009)
26. Dy, J.G., Brodley, C.E.: Feature Subset Selection and Order Identification for Unsupervised Learning. In: ICML 2000, pp. 247–254 (2000)
27. Law, M.H.C., Jain, A.K., Figueiredo, M.A.T.: Feature Selection in Mixture-Based Clustering. In: NIPS 2002, pp. 625–632 (2002)
28. Roth, V., Lange, T.: Feature Selection in Clustering Problems. In: NIPS 2003 (2003)
29. Jennifer, G.D., Brodley, C.E., Kak, A.C., Broderick, L.S., Aisen, A.M.: Unsupervised Feature Selection Applied to Content-Based Retrieval of Lung Images. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 25(3), 373–378 (2003)
30. Jennifer, G.D., Brodley, C.E.: Feature Selection for Unsupervised Learning. Journal of Machine Learning Research (JMLR) 5, 845–889 (2004)
31. Law, M.H.C., Figueiredo, M.A.T., Jain, A.K.: Simultaneous Feature Selection and Clustering Using Mixture Models. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 26(9), 1154–1166 (2004)
32. Boutemedjet, S., Ziou, D., Bouguila, N.: Unsupervised Feature Selection for Accurate Recommendation of High-Dimensional Image Data. In: NIPS 2007 (2007)
33. Boutsidis, C., Mahoney, M.W., Drineas, P.: Unsupervised feature selection for principal components analysis. In: KDD 2008, pp. 61–69 (2008)
34. Mitra, P., Murthy, C.A., Pal, S.K.: Unsupervised FeatureSelection Using Feature Similarity. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 24(3), 301–312 (2002)
35. Yu, L., Liu, H.: Efficient Feature Selection via Analysis of Relevance and Redundancy. Journal of Machine Learning Research (JMLR) 5, 1205–1224 (2004)
36. He, X., Cai, D., Niyogi, P.: Laplacian Score for Feature Selection. In: NIPS 2005 (2005)
37. Zhao, Z., Liu, H.: Spectral feature selection for supervised and unsupervised learning. In: ICML 2007, pp. 1151–1157 (2007)
38. Nie, F., Xiang, S., Jia, Y., Zhang, C., Yan, S.: Trace Ratio Criterion for Feature Selection. In: AAAI 2008, pp. 671–676 (2008)
39. Cai, D., Zhang, C., He, X.: Unsupervised feature selection for multi-cluster data. In: KDD 2010, pp. 333–342 (2010)
40. Zhao, Z., Wang, L., Liu, H.: Efficient Spectral Feature Selection with Minimum Redundancy. In: AAAI 2010 (2010)
41. Jiang, Y., Ren, J.: Eigenvalue Sensitive Feature Selection. In: ICML 2011 (2011)

Restricted Deep Belief Networks for Multi-view Learning

Yoonseop Kang[1] and Seungjin Choi[1,2]

[1] Department of Computer Science
[2] Division of IT Convergence Engineering
Pohang University of Science and Technology
San 31 Hyoja-dong, Nam-gu, Pohang 790-784, Korea
{e0en,seungjin}@postech.ac.kr

Abstract. Deep belief network (DBN) is a probabilistic generative model with multiple layers of hidden nodes and a layer of visible nodes, where parameterizations between layers obey harmonium or restricted Boltzmann machines (RBMs). In this paper we present *restricted deep belief network* (RDBN) for multi-view learning, where each layer of hidden nodes is composed of *view-specific* and *shared* hidden nodes, in order to learn individual and shared hidden spaces from multiple views of data. View-specific hidden nodes are connected to corresponding view-specific hidden nodes in the lower-layer or visible nodes involving a specific view, whereas shared hidden nodes follow inter-layer connections without restrictions as in standard DBNs. RDBN is trained using layer-wise contrastive divergence learning. Numerical experiments on synthetic and real-world datasets demonstrate the useful behavior of the RDBN, compared to the multi-wing harmonium (MWH) which is a two-layer undirected model.

1 Introduction

Multi-view learning refers to methods for learning from examples that have multiple (independent or dependent) representations, each of which may arise from different views such as modalities or sensors. For instance, in web page classification, one view describes a web page using the text appearing on the document itself, while the other view leads to the anchor text to hyperlinks pointing to this page from other pages [2]. In multimedia mining, images are described by color histograms (one view) and annotated text (the other view), so it is desirable to exploit these two information sources together to boost the performance of image classification [18]. In brain wave classification where EEG data are measured from multiple subjects who undergo the same mental task, each view corresponds to each subject, then learning shared latent spaces provides useful semantic features for EEG classification [11].

Learning latent spaces that capture the relevant information shared by multiple views of data lies at the heart of multi-view learning, especially when views are dependent. One of the oldest but the most popular method is canonical correlation analysis (CCA) [8] which identifies linear relationships between two sets

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 130–145, 2011.

of observations. The shared Gaussian process latent variable model (sGPLVM) [14] is a nonlinear extension of CCA, in which Gaussian process regression is used to learn common hidden structure shared between corresponding sets of heterogenous observations. Manifold integration [4] combines similarity matrices (each of which is determined by a specific view) into a compromise matrix that faithfully reflects multiple sensory information.

Most of these methods assume that views are fully independent conditioned on common latent variables, i.e., only shared latent variables are considered to explain the dependency between views. However, in real-world problems, this assumption is not satisfied. Thus it was suggested that separate latent spaces are learned to model the shared characteristics across views and individual components of observations [11,13]. Group nonnegative matrix factorization (GNMF) learn jointly individual latent spaces and shared latent spaces, decomposing nonnegative data matrix into a product of two nonnegative factor matrices where the basis matrix is composed of common and individual basis vectors [11]. Factorized orthogonal latent space (FOLS) method [13] factorizes the latent space into shared and private latent spaces, enforcing orthogonality between them. These aforementioned methods have limitations since CCA and GNMF are linear techniques and FOLS requires large memory storage and expensive time complexity to handle operations involving a big Gram matrix, which is not scalable as the number of samples increases.

Multi-wing harmonium (MWH) is a two-layer undirected graphical model designed to handle multiple view data. MWH consists of two or more wings where each wing takes the input associated with a single view, extending harmoniums [16] and restricted Boltzmann machines (RBMs) [5]. Harmoniums were also extended to the exponential family [17]. MWH inherits the advantages of exponential family harmoniums (EFHs) such as easy inference and distributed representations over latent variables (see [17] for other advantages). Deep belief network (DBN) is composed of RBMs with multiple layers of hidden nodes, which is trained efficiently in layer-wise manner based on contrastive divergence [5,6]. See [1] for excellent tutorial on DBNs. Multilayer structure of DBNs allow more complex representation of data than RBMs. However, unlike other Boltzmann machine-based models, MWHs cannot be naturally extended to form a deep network.

In this paper we present *restricted deep belief network* (RDBN) for partially correlated multiple view data. We first present a modification of EFHs where view-specific hidden nodes are restricted to have undirected connections to only visible nodes involving corresponding views whereas shared hidden nodes are connected to all the visible nodes. This model inherits advantages of FOLS and MWH simultaneously. The model can efficiently evaluate latent variables as MWH, while still being capable of modelling shared and private information separately.

Then we stack these modified harmoniums to construct RDBN in such a way that view-specific hidden nodes are connected to corresponding view-specific hidden nodes in the lower-layer and shared hidden nodes are connected to all the

hidden nodes in the lower-layer. We train RDBNs using layer-wise contrastive divergence learning, to learn view-specific and shared hidden spaces from multiple views. Numerical experiments on a synthetic dataset, NORB-small dataset, and ESL photo dataset demonstrated the useful behavior of RDBN, compared to MWHs and its direct multilayer version.

2 Related Work

2.1 Exponential Family Harmonium

Most of graphical models including with a layer of hidden nodes and other layer of observed nodes are based on a directed graph. These directed two-layer models allows easy sampling of visible layer, and easy handling of latent variables. However, it is often very difficult and time consuming to calculate posterior distribution of hidden nodes given visible nodes. For the tasks that requires fast evaluation of latent variable given a test sample, directed models would not be very effective.

EFH is a two-layer probabilistic graphical model whose probability distributions are constrained to be exponential family distribution. EFH consists of a set of hidden nodes $\boldsymbol{h}$, and a set of visible nodes $\boldsymbol{v}$ connected by undirected edges. By incorporating undirected connections, the model calculates posterior distribution of hidden nodes $p(\boldsymbol{h}|\boldsymbol{v})$ much faster than directed models, making it more suitable for the tasks including document searching and automatic image annotation.

To define an EFH, we start from choosing marginal distributions for visible nodes $\boldsymbol{v}$ and hidden nodes $\boldsymbol{h}$. As the distributions are constrained to be exponential family distributions, the marginal distributions are defined as below:

$$p(\boldsymbol{v}) = \prod_i \exp\Big\{\sum_a \lambda_{ia} f_{ia}(x_i) - A_i(\{\lambda_{ia}\})\Big\}, \tag{1}$$

$$p(\boldsymbol{h}) = \prod_j \exp\Big\{\sum_c \eta_{jc} g_{jc}(h_j) - B_j(\{\eta_{jc}\})\Big\}, \tag{2}$$

where f_{ia} and g_{jc} are a, and cth sufficient statistics of v_i and h_j. λ^x and η are parameters, and A and B are log partition functions. Note that each v_is and h_j are assumed to be independent to each other.

With this definition, we define joint distribution of $\boldsymbol{v}$ and $\boldsymbol{h}$ by combining their distributions multiplicatively. The joint distribution is defined by introducing quadratic terms to inter-layer relationship:

$$p(\boldsymbol{v}, \boldsymbol{h}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda_{ia} f_{ia}(v_i) + \sum_{j,c} \eta_{jc} g(h_j) + \sum_{i,j,a,c} W_{ijac} f_{ia}(v_i) g_{jb}(h_j)\Big\}. \tag{3}$$

As the model is a bipartite graph, between-layer conditional distributions can be represented as products of distributions of individual nodes. The conditional

distributions are derived as below:

$$p(\boldsymbol{v}|\boldsymbol{h},\theta) = \prod_i \exp\Big\{\sum_a \hat{\lambda}_{ia} f_{ia}(v_i) - A_i(\{\hat{\lambda}_{ia}\})\Big\}, \tag{4}$$

$$p(\boldsymbol{h}|\boldsymbol{v},\theta) = \prod_j \exp\Big\{\sum_c \hat{\eta}_{jc} g_{jc}(h_j) - B_j(\{\hat{\eta}_{jc}\})\Big\}, \tag{5}$$

where shifted parameters are

$$\hat{\lambda}_{ia} = \lambda_{ia} + \sum_{j,c} W_{ijac} g_{jc}(h_j), \tag{6}$$

$$\hat{\eta}_{jc} = \eta_{jc} + \sum_{i,a} W_{ijac} f_{ia}(x_i). \tag{7}$$

As the conditional distribution of nodes are independent to each other, sampling posterior distribution of hidden nodes is done by just simply evaluating conditional distribution $p(\boldsymbol{h}|\boldsymbol{v},\theta)$. An EFH with binary distributions for $\boldsymbol{v}$ and $\boldsymbol{h}$ becomes equivalent to a RBM.

2.2 Multi-Wing Harmonium

Multi-wing harmonium (MWH) [18] models joint distribution of multi-view data using two-layer graphical model. Given two-view data, MWH uses two sets of visible nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ connected to hidden nodes $\boldsymbol{h}$. By choosing different distribution for different type of information, the model achieves better representation of data.

Construction of an MWH is similar to the one of EFH. The only difference is that we split visible nodes $\boldsymbol{v}$ to two sets $\boldsymbol{x}$ and $\boldsymbol{y}$. First we choose marginal distributions for $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{h}$ as below:

$$p(\boldsymbol{x}) = \prod_i \exp\Big\{\sum_a \lambda^x_{ia} f^x_{ia}(x_i) - A^x_i(\{\lambda^x_{ia}\})\Big\}, \tag{8}$$

$$p(\boldsymbol{y}) = \prod_k \exp\Big\{\sum_b \lambda^y_{kb} f^y_{kb}(y_k) - A^y_k(\{\lambda^y_{kb}\})\Big\}, \tag{9}$$

$$p(\boldsymbol{h}) = \prod_j \exp\Big\{\sum_c \eta_{jc} g_{jc}(h_j) - B_j(\{\eta_{jc}\})\Big\}, \tag{10}$$

where f^x_{ia} and f^y_{kb} are a and bth sufficient statistics of x_i and y_k. λ^x, λ^y and η are parameters, and A^x, A^y, and B are log partition functions. Given marginal distribution, joint distribution of nodes is defined straightforwardly:

$$p(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) + \sum_{j,c} \eta_{jc} g(h_j) + \sum_{i,j,a,c} W^x_{ijac} f^x_{ia}(x_i) g_{jb}(h_j) + \sum_{k,j,b,c} W^y_{kjbc} f^y_{kb}(x_y) g_{jc}(h_j)\Big\}. \tag{11}$$

The conditional distributions are derived as below:

$$p(\boldsymbol{x}|\boldsymbol{h},\theta) = \prod_i \exp\Big\{\sum_a \hat{\lambda}^x_{ia} f^x_{ia}(x_i) - A^x_i(\{\hat{\lambda}^x_{ia}\})\Big\}, \tag{12}$$

$$p(\boldsymbol{y}|\boldsymbol{h},\theta) = \prod_k \exp\Big\{\sum_b \hat{\lambda}^y_{kb} f^y_{kb}(y_k) - A^y_k(\{\hat{\lambda}^y_{kb}\})\Big\}, \tag{13}$$

$$p(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{y},\theta) = \prod_j \exp\Big\{\sum_c \hat{\eta}_{jc} g_{jc}(h_j) - B_j(\{\hat{\eta}_{jc}\})\Big\}, \tag{14}$$

with shifted parameters

$$\hat{\lambda}^x_{ia} = \lambda^x_{ia} + \sum_{j,c} W^x_{ijac} g_{jc}(h_j), \tag{15}$$

$$\hat{\lambda}^y_{kb} = \lambda^y_{kb} + \sum_{j,c} W^y_{kjbc} g_{jc}(h_j), \tag{16}$$

$$\hat{\eta}_{jc} = \eta_{jc} + \sum_{i,a} W^x_{ijac} f_{ia}(x_i) + \sum_{k,b} W^y_{kjbc} f_{jb}(y_j). \tag{17}$$

The model can be extended to the case of more than two sets of visible nodes. With a single set of visible nodes, the model shrinks down to a EFH or RBM. To enhance discriminative performance of MWH, labeled version [19] and large-margin approach [3] were also proposed.

3 Restricted DBNs

3.1 Multi-view Harmonium

Many multi-view algorithms are based on a weak assumption that all views are completely correlated. MWH also assumes that views are completely independent when the values of hidden nodes are given. However, the views are often incompletely correlated on real-world datasets. On these datasets, MWH will mix view-specific information with shared information and fail to obtain optimal representation of data. To overcome this problem, we need to model the view-specific information and shared information seperately.

Recent multi-view learning algorithms including work of Salzmann et al.[13] learns from partially independent multi-view data sets by seperating view-specific latent variables from latent variables shared among views.

However, the limitation of existing models is evident. For example, as the FOLS framework requires a Gram matrix, so the model consumes memory storage proportional to N^2, where N is the number of training samples. The evaluation also requires computing time of complexity $O(N)$, as we need to calculate kernel function for every training sample and a test sample. Moreover, existing models including FOLS and group NMF requires additional parameters to control orthogonality between view-specific and shared latent spaces, and these parameters should be selected by going through computationally expensive procedures including cross-validation or by hand.

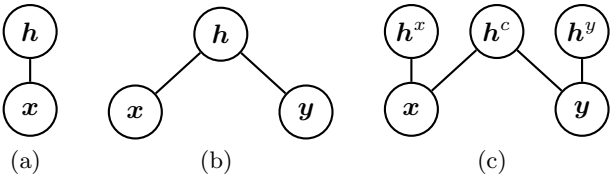


Fig. 1. Graphical models of (a) EFH, (b) MWH, and (c) multi-view harmonium. Repetitions of variables are not denoted for simplicity.

Taking advantage of fast learning and inference of harmonium models and the idea of seperating shared and common latent variables, we extend MWH to devise a new model named *multi-view harmonium*.

This model incorporates view-specific hidden nodes $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ in addition to common hidden nodes $\boldsymbol{h}^c$, to model view-specific, uncorrelated information. $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ are only connected to their corresponding set of visible nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ with connection weights $\boldsymbol{U}^x$ and $\boldsymbol{U}^y$. Graphical representations show the difference between other models and our model (Fig. 1). Joint probability of visible and hidden nodes is as below:

$$
\begin{aligned}
&p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}^x, \boldsymbol{h}^y, \boldsymbol{h}^c | \theta) \propto \\
&\exp\Big\{ \sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) + \sum_{j,c} \eta^c_{jc} g^c(h^c_j) + \sum_{m,d} \eta^x_{md} g^x(h^x_m) \\
&+ \sum_{n,e} \eta^y_{ne} g^y(h^y_n) + \sum_{i,j,a,c} W^x_{ijac} f^x_{ia}(x_i) g_{jb}(h_j) + \sum_{k,j,b,c} W^y_{kjbc} f^y_{kb}(y_k) g_{jc}(h_j) \\
&+ \sum_{i,m,a,d} U^x_{imad} f^x_{ia}(x_i) g^x_{md}(h^x_m) + \sum_{k,n,b,e} U^y_{knbe} f^y_{kb}(y_k) g^y_{ne}(h_n) \Big\}. \qquad (18)
\end{aligned}
$$

Conditional distributions can easily be derived from this joint distribution.

Depending on the type of data, we can choose appropriate distributions from exponential-family. To handle continuous-valued inputs, one can use Gaussian distribution for visible nodes $\boldsymbol{x}$ [7], where the conditional probability is defined as:

$$
p(x_i | \boldsymbol{h}^x, \boldsymbol{h}^c, \theta) = \mathcal{N}(x_i | \boldsymbol{W}^x_i \boldsymbol{h}^c + \boldsymbol{U}^x_i \boldsymbol{h}^x + b^x_i, 1), \qquad (19)
$$

assuming x_i has zero mean and unit variance. Rectified linear units(ReLU) for hidden nodes is also helpful in handling continuous values [12], where value of hidden node h_j given visible node $\boldsymbol{x}$ is defined as:

$$
h_j = \max(0, \sum_i W_{ij} x_i + h_j + \mathcal{N}(0, 1)). \qquad (20)
$$

When modeling term occurence on bag-of-words representation, we can use Poisson distribution:

$$
p(x_i | \boldsymbol{h}^x, \boldsymbol{h}^c, \theta) = \text{Poission}(x_i | \exp(\alpha_i + \boldsymbol{W}^x_i \boldsymbol{h}^x + \boldsymbol{U}^x_i \boldsymbol{h}^c)). \qquad (21)
$$

We train multi-view harmonium by maximizing log-likelihood. Given the joint distribution $p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}^x, \boldsymbol{h}^y, \boldsymbol{h}^c|\theta)$ we can integrate out the hidden units to obtain likelihood function:

$$p(\boldsymbol{x}, \boldsymbol{y}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) - \sum_j B^c(\{\hat{\eta}^c_{jc}\}) - \sum_m B^x(\{\hat{\eta}^x_{md}\}) - \sum_n B^y(\{\hat{\eta}^y_{ne}\})\Big\}, \tag{22}$$

where B^x, B^y and B^c are log-partition functions of marginal distribution of hidden variables $\boldsymbol{h}^c$, $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ and the shifted parameters are

$$\hat{\eta}^c_{jc} = \eta^c_{jc} + \sum_{i,a} W^x_{ijac} f^x_{ia}(x_i) + \sum_{k,b} W^y_{kjbc} f^y_{kb}(y_j), \tag{23}$$

$$\hat{\eta}^x_{md} = \eta^x_{md} + \sum_{i,a} U^x_{imad} f^x_{ia}(x_i), \tag{24}$$

$$\hat{\eta}^y_{ne} = \eta^y_{ne} + \sum_{k,n,b,e} U^y_{knbe} f^y_{kb}(y_k). \tag{25}$$

To maximize expected log-likelihood over data distribution $\mathcal{L} = \langle \log p(\boldsymbol{x}, \boldsymbol{y}|\theta)\rangle_{\tilde{p}}$, we can use gradient ascent to update parameters θ. The derivatives on parameters related to $\boldsymbol{x}$ are given as follows:

$$\frac{\partial \mathcal{L}}{\partial W^x_{ijac}} = \langle f_{ia}(x_i) B'_{jc}(\hat{\eta}^c_{jc})\rangle_{\tilde{p}} - \langle f_{ia}(x_i) B'_{jc}(\hat{\eta}^c_{jc})\rangle_p, \tag{26}$$

$$\frac{\partial \mathcal{L}}{\partial U^x_{imad}} = \langle f_{ia}(x_i) B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_{\tilde{p}} - \langle f_{ia}(x_i) B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_p, \tag{27}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda^x_{ia}} = \langle f^x_{ia}(x_i)\rangle_{\tilde{p}} - \langle f^x_{ia}(x_i)\rangle_p, \tag{28}$$

$$\frac{\partial \mathcal{L}}{\partial \eta^c_{jc}} = \langle B^{c'}_{jc}(\hat{\eta}^c_{jc})\rangle_{\tilde{p}} - \langle B^{c'}_{jc}(\hat{\eta}^c_{jc})\rangle_p, \tag{29}$$

$$\frac{\partial \mathcal{L}}{\partial \eta^x_{md}} = \langle B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_{\tilde{p}} - \langle B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_p. \tag{30}$$

where $\langle\cdot\rangle_{\tilde{p}}$ and $\langle\cdot\rangle_p$ are expectation over data distribution and model distribution, and $B^{c'}_{jc} = \partial B^c_{jc}/\partial \eta^c_{jc}$ and $B^{x'}_{md} = \partial B^x_{md}/\partial \eta^x_{md}$ are partial derivatives of log-partition functions. Derivatives related to $\boldsymbol{y}$ can be derived similarly. To calculate the expectations from data and model distribution, we need samples from the distributions. However, as we cannot directly sample from the joint distribution of visible nodes and hidden nodes, we use Gibbs sampling and sample from conditional distributions in each Gibbs steps. Values of each node can be sampled from conditional distribution in a single step when other layer's nodes are given.

However, we need infinite steps of Gibbs sampling to calculate exact model distribution. To avoid the problem, we may approximate log-likelihood by using contrastive divergence (CD) learning instead [5]. The key idea of CD learning is

Algorithm 1. Training multi-view harmonium using CD learning with mini-batch size K and M Gibbs steps

Input: $\mathcal{D} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}$
Output: $\theta = \{\boldsymbol{W}^x, \boldsymbol{W}^y, \boldsymbol{U}^x, \boldsymbol{U}^y, \eta^x, \eta^y, \eta^c, \lambda^x, \lambda^y\}$
procedure CDLEARNING($\mathcal{D}$)
    Randomly initialize parameters θ.
    repeat
        Pick K samples as a mini-batch
        repeat
            Sample $\tilde{\boldsymbol{h}}^x \sim p(\boldsymbol{h}^x|\boldsymbol{x}_t, \theta)$, $\tilde{\boldsymbol{h}}^y \sim p(\boldsymbol{h}^y|\boldsymbol{y}_t, \theta)$.
            Sample $\tilde{\boldsymbol{h}}^c \sim p(\boldsymbol{h}^c|\boldsymbol{x}_t, \boldsymbol{y}_t, \theta)$.
            repeat
                Sample $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{h}^x, \boldsymbol{h}^c, \theta)$.
                Sample $\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{h}^y, \boldsymbol{h}^c, \theta)$.
                Sample $\boldsymbol{h}^x \sim p(\boldsymbol{h}^x|\boldsymbol{x}, \theta)$.
                Sample $\boldsymbol{h}^y \sim p(\boldsymbol{h}^y|\boldsymbol{y}, \theta)$.
                Sample $\boldsymbol{h}^c \sim p(\boldsymbol{h}^c|\boldsymbol{x}, \boldsymbol{y}, \theta)$.
            until M steps
        until for all $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ in mini-batch.
        update θ using gradient ascent on $\mathcal{L}$.
    until θ is converged
end procedure

to initialize Gibbs chain to the training data, and to run just a few Gibbs steps instead of infinite steps to approximate model distribution. Even running just a single Gibbs step is known to work well in practice.

Training harmoniums using CD learning requires a gradient calculated over whole training set, and this is often a time consuming task. Instead, we can pick mini-batches from training set, and calculate gradient over mini-batches to save time and space required for training. Summary of procedure for training multi-view harmonium using CD learning is shown in Algorithm 1.

3.2 Restricted DBN

Introducing view-specific hidden nodes brings our model capability of modeling partial correlation, but view-specific hidden nodes also enables us to easily extend our model to form a deep network. Values of view specific hidden layer nodes will be fed as data for upper layer of deep network (Fig. 2). By forming deep network in this manner, the information considered to be "uncorrelated" by limited representation power of current layer will be sent to upper layer and view-to-view correlation will be further analyzed.

Training a deep network using back-propagation is known to be often inefficient. Therefore, training an RDBN is done in greedy, layer-wise manner [6]. We train each layer of multi-view harmoniums using algorithm 1, starting from the bottom layer. After training each layer is finished, a harmonium on the next layer uses samples of view-specific hidden nodes from the trained harmonium

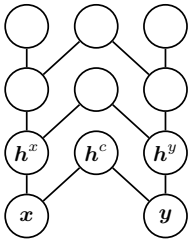


Fig. 2. The graphical model of RDBN. View-specific hidden nodes act as new visible nodes for upper layer of the deep network model.

as its training set. Training of RDBN ends when the procedure reaches the top layer.

3.3 Inferring One View from the Other

Inferring values of unobserved views from other observed views on RDBN is not as easy as inferring the values from two-layer MWH. Instead of using Variational approximation, we choose to perform Gibbs sampling in a systematic manner. The procedure can be divided into three steps, and each step is executed layer by layer (Fig. 3).

1. In the first step, we start from the bottom layer. Given observed values $\boldsymbol{x}$, the hidden nodes of observed view $\boldsymbol{h}^x$ are only nodes that whose probability can be exactly calculated. Therefore we sample the hidden nodes $\boldsymbol{h}^x$ and proceed to the upper layer to repeat this procedure (Fig. 3-b).
2. When the first step reaches the top layer, we run Gibbs sampling for top two layers, while nodes of observed views $\boldsymbol{v}^x$, $\boldsymbol{h}^x$ are fixed to values determined on the first step. In other words, we sample from $p(\boldsymbol{v}^y, \boldsymbol{h}^c, \boldsymbol{h}^y | \boldsymbol{v}^x, \boldsymbol{h}^x, \theta)$. Then we get the values for top two layers of unobserved views $\boldsymbol{h}^c$, $\boldsymbol{v}^y$. We use average of repeatedly sampled values for the unobserved nodes (Fig. 3-c).
3. Finally, we move to lower layers. Given observed nodes $\boldsymbol{x}$ and $\boldsymbol{h}^x$, and hidden units of unobserved views sampled in previous steps $\boldsymbol{h}^y$, we sample from $p(\boldsymbol{y}, \boldsymbol{h}^c | \boldsymbol{x}, \boldsymbol{h}^x, \boldsymbol{h}^y, \theta)$. Repeating the process until we reach bottom layer gives us the values of unobserved views (Fig. 3-d).

When common hidden nodes $\boldsymbol{h}^c$ of bottom layer of deep network are binary logistic nodes, we can skip Gibbs sampling and directly evaluate $p(\boldsymbol{y}|\boldsymbol{x}, \theta)$ up to a normalization constant by rearranging terms:

$$p(\boldsymbol{y}|\boldsymbol{x}, \theta) \propto \prod_j \Big[1 + \exp\Big\{\sum_{i,a}(W^x_{ija} f^x_a(x_i) + \lambda^x_{ia} f^x_a(x_i)) + \sum_{k,b}((W^y_{kjb} + U^y_{knb}) f^y_b(y_k) + \lambda^y_{kb} f^y_b(y_k)) + c^c_j\Big\}\Big], \tag{31}$$

For other layers of RDBN, we use Gibbs sampling instead. Empirically, about 10 times of Gibbs sampling for each layer gave a fair result.

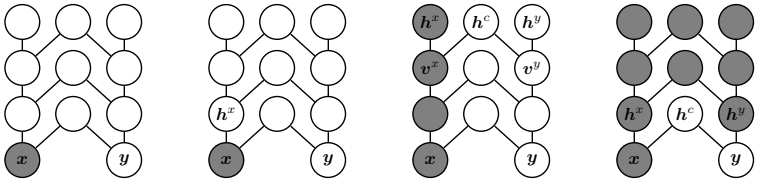


Fig. 3. Inferring procedure of RDBN. Variables with known values at current step is marked as a shaded circle.

4 Numerical Experiments

In this section, we compare our model to MWH. We also compare our method with FOLS-GPLVM [13], and shared GPLVM(SGPLVM) [14], which are directed, non-parametric multi-view latent variable models based on Gaussian process latent variable model [9]. We considered CCA as a baseline too. On a synthetic dataset, we show the effect of modelling view-specific information. Then we run experiments on widely-used NORB-small dataset, and ESL photo dataset in a view-to-view conversion task to numerically compare performance of RDBN and MWH and GPLVM-based models. We used Gibbs sampling for inference on the graphical models.

4.1 Synthetic Example

To show the effectiveness of additional hidden nodes, we performed an experiment taken from recent work of Salzmann et al. [13]. We constructed a synthetic partially correlated multi-view dataset. To illustrate common and view-specific latent variables, we used sinusoidal functions of $\boldsymbol{t}$ with different phases and frequencies:

$$\boldsymbol{x} = \sin(2\pi\boldsymbol{t}), \quad \boldsymbol{z}^1 = \cos(\pi\pi\boldsymbol{t}), \quad \boldsymbol{z}^2 = \sin(\sqrt{5}\pi\boldsymbol{t}),$$
$$\boldsymbol{m}^1 = [\boldsymbol{x}, \boldsymbol{z}^1], \quad \boldsymbol{m}^2 = [\boldsymbol{x}, \boldsymbol{z}^2].$$

We ramdomly projected $\boldsymbol{m}^1$ and $\boldsymbol{m}^2$ to 20 dimensional space and added independent Gaussian noise of variance 0.01 and correlated noise $0.02\sin(3.6\pi\boldsymbol{t})$ to obtain our final multi-view synthetic dataset.

A multi-view harmonium, which is a two-layer RDBN, with 1 common hidden node and 1 view-specific hidden node is used, and we used a MWH and CCA for comparison. 3 hidden nodes for MWH and projection to 3-dimensional space for CCA was chosen for fair comparison. We used Gaussian nodes and ReLU for visible and hidden nodes.

Training was done using 4000 training samples and we calculated hidden node activations from 500 test samples. We checked the correspondence between ground-truth latent variables, and obtained hidden node activations.

MWH, SGPLVM and CCA failed to infer latent variables correctly. No hidden node activations corresponded to latent variables used to generate data. In the

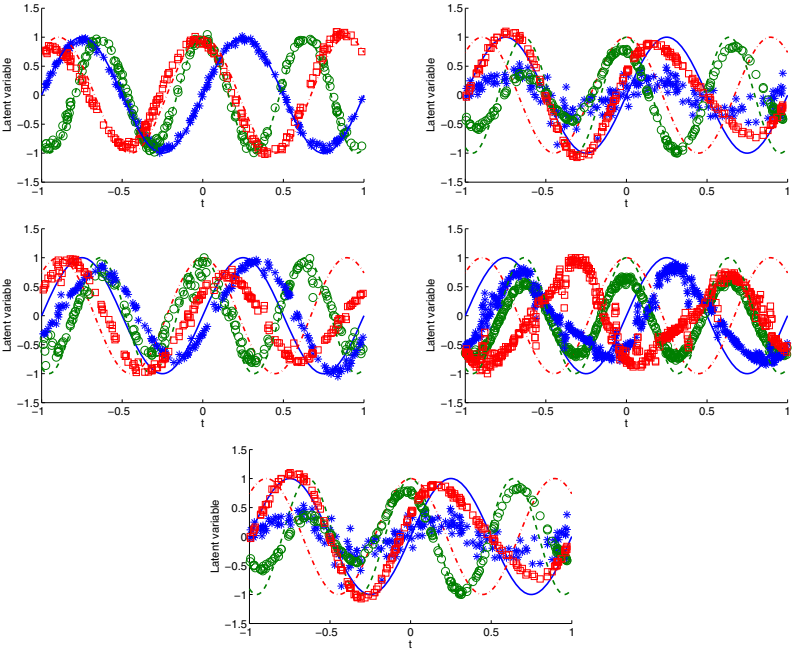


Fig. 4. Latent variables generated by muti-view harmonium , MWH, FOLS, Shared GPLVM and CCA (starting from top-left to bottom-right). Outputs of methods are normalized to have zero mean and maximum absolute value to be 1. Latent variables used to generate data are depicted as lines: Blue solid line: $\boldsymbol{x}$, green dotted line: $\boldsymbol{z}^1$, and red dash-dotted line: $\boldsymbol{z}^2$. Each dimension of outputs are depicted as markers with different colors and shapes.

other hand, our model, multi-view harmonium, found common and view-specific latent variables by its hidden node activations [4]. Common hidden nodes and view-specific hidden node activations exactly corresponded to the common and view-specific latent variables. Although FOLS-GPLVM also discovered latent variables correctly, the result was not as accurate as the result of our model. Moreover, our model was able to separate common and view-specific information without any help of additional scheme, while FOLS had to minimize mutual information explicitly.

4.2 Object Conversion on NORB-Small

Training set of NORB-small dataset [10] contains 24,300 stereo images of 5 generic categories of objects on a white background, imaged under various lighting conditions and camera angles. Each category contains 5 instances of objects. For example, the category 'car' contains pictures of 5 different cars.

To evaluate view-to-view conversion performance, we constructed a multi-view dataset by taking pairs of images with same lighting and camera angles, but different category of objects (airplanes - cars), that means each sample in

Fig. 5. Graphical representation of view to view conversion task on NORB-small dataset

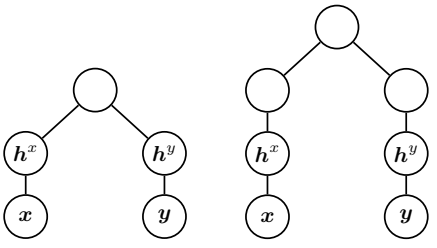


Fig. 6. Graphical model of 3 and 4-layer MWHs used in experiments on NORB-small dataset and ESL dataset

the dataset contains a image of airplane and a car, taken with same condition. 1200 image pairs were used as a training set and 3600 image pairs were used as a test set. After training, model was asked to generate a car image with same imaging condition from a given airplane image (Fig. 5). Images were resized to the size 32×32, and pixels on each position of an image were normalized to have zero mean and variance 1. The variance of pixel values was calculated across whole training set on the same position on images. The root mean squared error (RMSE) was computed:

$$\text{RMSE} = \sqrt{\frac{1}{N_{samples}} \sum_t \frac{1}{N_{pixels}} ||\boldsymbol{x}_t^{truth} - \boldsymbol{x}_t^{rec}||^2}, \tag{32}$$

where $N_{samples}$ and N_{pixels} are the number of samples and of pixels, respectively, and $\boldsymbol{x}_t^{truth}$ and $\boldsymbol{x}_t^{rec}$ are ground-truth and reconstructed images, respectively.

For our experiment, we constructed 2, 3, and 4-layer RDBN. For two layers from the bottom, we used Gaussian and ReLU, to handle continuous valued data. Logistic binary nodes were used for remaining two layers.

We also constructed a deep network model for MWH by attaching harmoniums under each views of MWH for a fair comparison with RDBN. For example, 4-layer MWH was constructed by attaching additional two layers of Harmoniums under each view of a MWH (Fig. 6). Inferring an unobserved view was done by the procedure below.

1. Sample nodes of observed view layer by layer from the bottom.
2. Infer the unobserved view of MWH on the top layer.
3. Sample unobserved nodes, from top to bottom layer.

Table 1. Averaged RMS reconstruction errors of RDBN, MWH models and random reconstruction in the view-to-view conversion task on NORB-small dataset

Method	RMSE	Method	RMSE
4-layer RDBN	**0.0923**	4-layer MWH	0.1379
3-layer RDBN	0.0936	3-layer MWH	0.1377
2-layer RDBN	0.0958	2-layer MWH	0.0973
FOLS-GPLVM	0.2489	SGPLVM	0.2089
Random	0.5205		

We also added additional hidden nodes to MWH to ensure same number of parameters are used. For example, if a RDBN had 200 view-specific hidden nodes and 800 common hidden nodes, each view of corresponding layer of a MWH had $200 + 800 = 1000$ hidden nodes. We also tried 2-layer and 3-layer RDBN to see the effect of number of layers. Each network was trained 200 iterations with learning rate of 0.001. Instead of full batch learning, we used mini-batches with 100 samples. SGPLVM and FOLS-GPLVM were trained with 200 iterations, and their latent dimension was automatically decided by a heuristic method of Neil Lawrence's software. As a baseline, we used images with random pixel values as reconstruction images and compared the reconstruction errors with the results of RDBNs and MWH models.

The experimental result showed remarkable difference between RDBN and MWH algorithm. 4-layer RDBN showed the smallest error than any configuration of MWH, FOLS-GPLVM and SGPLVM. Moreover, adding each additional layer to RDBN reduced reconstruction error, while adding layer to MWH damaged the performance of the model (Table 1).

4.3 Image Annotation on ESL Photo Dataset

To examine the capability of our deep model to find nontrivial relation between views, we applied our model to the task of image annotation. The experiment was done using ESL photo dataset [15], which contains 3464 photos collected from Flickr, annotated with 59 different tags. We calculated autocorrelogram of images with the setting of radius 6 and 32 colors on HSV color space. By this process we obtained a 192-dimensional feature vector for each image. We also calculated TF-IDF from our tag data. We used 2000 images for training, and 1044 images testing.

We compared 2, 3, 4-layer RDBNs, 2-layer MWH, sGPLVM, and FOLS-GPLVM. 30, 40, 60 hidden units were used for each hidden layer of RDBNs, and GPLVM models were trained with 200 iterations. Image autocorrelogram and TF-IDF of tags were assigned to each view of the models. On visible layer of the models, we used Gaussian nodes for images and Poisson nodes for tags. Each model was trained 500 iterations with learning rate 0.01. Again, we used mini-batches with 100 samples in training. We also tried annotating images with random tags as a baseline.

Precision-recall curve and mean average precision (MAP) were used as evaluation measure. MAP is defined as the mean of average precision (AP) over test

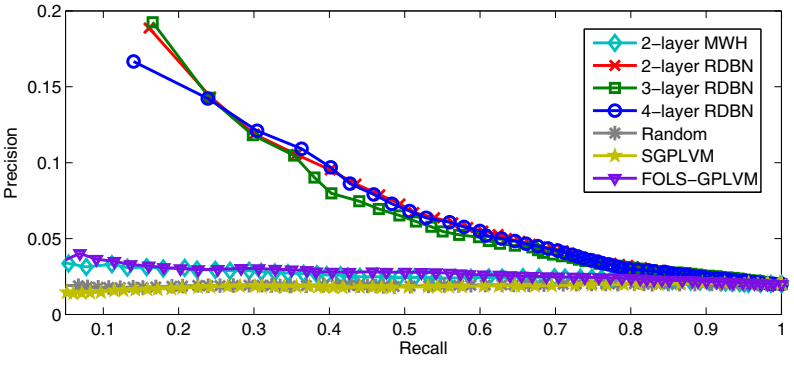


Fig. 7. Precision-recall curves for RDBNs, 2-layer MWH, SGPLVM, FOLS-GPLVM and random annotation. Vertical axis of the plot is trimmed for better comparison between results of RDBNs with different number of layers.

Table 2. The mean average precision of RDBNs, a MWH, SGPLVM, FOLS-GPLVM and random annotation in image annotation task on ESL dataset.

Method	MAP	Method	MAP
4-layer RDBN	**0.0620**	2-layer MWH	0.0251
3-layer RDBN	0.0598	SGPLVM	0.0180
2-layer RDBN	0.0565	FOLS-GPLVM	0.0267
Random	0.0190		

samples:

$$\mathrm{MAP} = \frac{\sum_t \mathrm{AP}(\boldsymbol{x}_t)}{N_{samples}} = \frac{1}{N_{samples}} \sum_t \sum_{j=1}^{N_{tags}} \mathrm{Precision}(\boldsymbol{x}_t, j), \tag{33}$$

where $N_{samples}$ and N_{tags} are the number of samples and of tags, respectively, and $\mathrm{Precision}(\boldsymbol{x}_t, j)$ denotes the precision when top j tags are chosen, given the query image $\boldsymbol{x}_t$. MAP is also calculated by area under precision-recall curve. We used this alternative definition of MAP to calculate the results reported here.

Precision-recall curve shows that RDBN algorithms clearly outperformed other methods, showing the advantage of modelling view-specific information seperately. Although the difference was not very significant, deep models showed higher average precision than 2-layer RDBN, showing the benefit of using deep models. In contrast, average precision of 2-layer MWH and SGPLVM were not very distinguishable from the result of random annotation (Table 2). Precision-recall curve also shows the difference between RDBNs and other models (Fig. 7).

5 Conclusions

In this paper, we have proposed the harmonium-based model that uses view-specific hidden nodes for multi-view data, to capture the view-specific

information of the data separated from shared information among views. Moreover, our model can be naturally extended to deep network to model more complex relationship between views by using view-specific hidden nodes as inputs to next layer. We have demonstrated the effectiveness of our approach by comparing our model to MWH, and their directed, non-parametric counterparts including SGPLVM and FOLS-GPLVM. the existing harmonium-based multi-view model, on various experiments on synthetic and real-world examples. And we showed significant improvement over other methods in the tested examples. In the future, we plan to investigate the modifications of RDBN model such as conditional RDBN to directly model conditional distribution between views. We also plan to extend our model to use label information for supervised and semi-supervised learning. Another future topic is extension of our model to time-series data or other structured data.

Acknowledgments. This work was supported by the Converging Research Center Program funded by the Ministry of Education, Science, and Technology (No. 2010K001171), NIPA ITRC Support Program (NIPA-2011-C1090-1131-0009), NIPA-MSRA Creative IT/SW Research Project, and NRF WCU Program (R31-10100).

References

1. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends® in Machine Learning 2(1), 1–127 (2009)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Annual Conference on Learning Theory (COLT), Madison, Wisconsin (1998)
3. Chen, N., Zhu, J., Xing, E.P.: Predictive subspace learning for multi-view data: A large margin approach. In: Advances in Neural Information Processing Systems (NIPS), vol. 23. MIT Press, Cambridge (2010)
4. Choi, H., Choi, S., Choe, Y.: Manifold integration with Markov random walk. In: Proceedings of the AAAI National Conference on Artificial Intelligence (AAAI), Chicago, IL (2008)
5. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
6. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
7. Hinton, G.E., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
8. Hotelling, H.: Relations between two sets of variates. Biometrika 28, 312–377 (1936)
9. Lawrence, N.: Gaussian process latent variable models for visualisation of high dimensional data. In: Advances in Neural Information Processing Systems (NIPS), vol. 16. MIT Press, Cambridge (2004)
10. LeCun, Y., Huang, F.J., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC (2004)

11. Lee, H., Choi, S.: Group nonnegative matrix factorization for EEG classification. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Clearwater Beach, Florida (2009)
12. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the International Conference on Machine Learning (ICML), Haifa, Israel (2010)
13. Salzmann, M., Ek, C.H., Urtasun, R., Darrell, T.: Factorized orthogonal latent spaces. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Sardinia, Italy (2010)
14. Shon, A.P., Grochow, K., Hertzmann, A., Rao, R.P.N.: Learning shared latent structure for image synthesis and robotic imitation. In: Advances in Neural Information Processing Systems (NIPS), vol. 17. MIT Press, Cambridge (2006)
15. Sinha, P., Jaini, R.: Classification and annotation of digital photos using optical context data. In: Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR), Niagara Falls, Canada (2008)
16. Smolensky, P.: Information processing in dynamical systems: Foundations of harmony theory. In: Rumelhart, D.E., McClelland, J.L. (eds.) Parallel Distributed Processing: Explorations in the Microstructures of Cognition: Foundations, vol. 1, pp. 194–281. MIT Press, Cambridge (1986)
17. Welling, M., Rosen-Zvi, M., Hinton, G.: Exponential family harmoniums with an application to information retrieval. In: Advances in Neural Information Processing Systems (NIPS), vol. 17. MIT Press, Cambridge (2005)
18. Xing, E.P., Yan, R., Hauptmann, A.G.: Mining associated text and images with dual-wing harmonium. In: Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI), Edinburgh, UK (2005)
19. Yang, J., Liu, Y., Xing, E.P., Hauptmann, A.G.: Harmonium models for semantic video representation and classification. In: Proceedings of the SIAM International Conference on Data Mining (SDM), Minineapolis, MN (2007)

Motion Segmentation by Model-Based Clustering of Incomplete Trajectories

Vasileios Karavasilis, Konstantinos Blekas, and Christophoros Nikou

Department of Computer Science, University of Ioannina,
PO Box1186, 45110 Ioannina, Greece
{vkaravas,kblekas,cnikou}@cs.uoi.gr

Abstract. In this paper, we present a framework for visual object tracking based on clustering trajectories of image key points extracted from a video. The main contribution of our method is that the trajectories are automatically extracted from the video sequence and they are provided directly to a model-based clustering approach. In most other methodologies, the latter constitutes a difficult part since the resulting feature trajectories have a short duration, as the key points disappear and reappear due to occlusion, illumination, viewpoint changes and noise. We present here a sparse, translation invariant regression mixture model for clustering trajectories of variable length. The overall scheme is converted into a Maximum A Posteriori approach, where the Expectation-Maximization (EM) algorithm is used for estimating the model parameters. The proposed method detects the different objects in the input image sequence by assigning each trajectory to a cluster, and simultaneously provides the motion of all objects. Numerical results demonstrate the ability of the proposed method to offer more accurate and robust solution in comparison with the mean shift tracker, especially in cases of occlusions.

Keywords: Motion segmentation, visual feature tracking, trajectory clustering, sparse regression.

1 Introduction

Visual target tracking is a preponderant research area in computer vision with many applications such as surveillance, targeting, action recognition from motion, motion-based video compression, teleconferencing, video indexing and traffic monitoring. Tracking is the procedure of generating inference about apparent motion given a sequence of images, where it is generally assumed that the appearance model of the target (e.g. color, shape, salient feature descriptors etc.) is known *a priori*. Hence, based on a set of measurements from image frames the target's position should be estimated.

Tracking algorithms may be classified into two main categories[1]: The first category is based on filtering and data association. It assumes that the moving object has an internal state, where the position of an object is estimated by combining the measurements with the model of the state evolution. Kalman

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 146–161, 2011.

filter [2] belongs to such methods which successfully tracks objects if the assumed type of motion is correctly modeled including cases of occlusion. Alternatively particle filters [3], including the condensation algorithm [4], are more general tracking methods without assuming specific type of densities. Finally there are methods relying on feature extraction and tracking using optical flow techniques [5]. A general drawback of this family of tracking algorithms is that the type of object's motion should be known and modeled correctly.

On the other hand, there are algorithms which are based on target representation and localization assuming for a probabilistic model for the object's appearance and the aim is to estimate it. More specifically, color or texture features of the object masked by an isotropic kernel are used to create a histogram. Then, the object's position is estimated by minimizing a cost function between the model's histogram and candidate histograms in the next image. A typical method in this category is the mean shift algorithm [1] and its extensions [6,7], where the object is supposed to be surrounded by an ellipse and the histogram is constructed from its internal pixel values. Also, algorithms based on the minimization of the differential Earth mover's distance [8,9] belong to this category.

Motion segmentation constitutes a significant application of tracking algorithms, which aims at identifying moving objects in a video sequence. It can be seen either as the post-processing step of a tracking algorithm, or as an assistive mechanism of the tracking algorithms by incorporating knowledge on the number of individual motions or their parameters. It has been considered in the framework of optical flow estimation, like in [10], where violations of brightness constancy and spatial smoothness assumptions caused by multiple motions are addressed and in [11], where affine flow is obtained by clustering the features into segments using the EM algorithm. Also, sparse features are clustered into groups and the number of groups is updated automatically over time in [12].

Trajectory clustering is also proposed in [13] where 3D trajectories are grouped using an agglomerative clustering algorithm and occlusions are handled by multiple tracking hypotheses. Finite mixtures of hidden Markov models (HMMs) were also employed [14] with parameter estimation obtained through the EM algorithm. Zappela *et al.* [15] project the space of the trajectories into a subspace of smaller dimensions and the clustering is performed by analyzing the eigenvalues of an affinity matrix, while in [16], overlapping trajectories are clustered and the resulting clusters are merged to cover large time spans.

Furthermore, spectral clustering approaches were also proposed, such as the method in [17], where the motions of the tracked feature points are modeled by linear subspaces and the approach in [18] where missing data from the trajectories are filled in by a matrix factorization method. Moreover, Yan *et al.* [19] estimate a linear manifold for every trajectory and then spectral clustering is employed to separate these subspaces. In [20], motion segmentation is accomplished by computing the shape interaction matrices for different subspace dimensions and combine them to form an affinity matrix that is used for spectral clustering.

Finally, many methods proposed independently rely on the separation of the image into layers. For example, in [21] tracking is performed in two stages: at

first foreground extracted blobs are tracked using graph cut optimization and then pedestrians are associated with blobs and their motion is estimated by a Kalman filter.

In this work, we present a framework for visual target tracking based on clustering trajectories of key points. The proposed method creates trajectories of image features that correspond to Harris corner features [22]. However, key point tracking introduces an additional difficulty because the resulting feature trajectories have a short duration, as the key points disappear and reappear due to occlusion, illumination and viewpoint changes. Therefore, we are dealing with time-series of variable length. Motion segmentation is converted next into a clustering of these input trajectories, in a sense of grouping together feature trajectories that belong to the same object. For this purpose, we use an efficient regression mixture model, which has three significant features: a) Sparseness, b) it is allowed to be translated in measurement space and c) its noise covariance matrix is diagonal and not spherical as in most cases. The above properties are incorporated through a Bayesian regression modeling framework, where the Expectation-Maximization (EM) algorithm can be applied for estimating the model parameters. Special care is given for initializing EM where an interpolating scheme is proposed based on executing successively the k-means algorithm over the duration of trajectories. Experiments show the robustness of our method to occlusions and highlight its ability to discover better the objects motion in comparison with the state-of-the art mean shift algorithm [1].

The rest of the paper is organized as follows: the procedure of feature extraction and tracking in order to create the trajectories is presented in section 2. The trajectories clustering algorithm is presented in section 3, experimental results are shown in section 4 and a conclusion is drawn in section 5.

2 Extracting Trajectories

Trajectories are constructed by tracking points in each frame of the video sequence. The main idea is to extract some salient points from a given image and associate them with points from previous images. To this end, we employ the so called *Harris corners* [22] and standard optical flow for the data association step [23]. Let us notice that any other scale or affine invariant features [24,25] would also be appropriate. In this work we use Harris corner features due to their simplicity, as they rely on the eigenvalues of the matrix of the second order derivatives of the image intensities.

Let T be the number of image frames and $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1,\ldots,N}$ be a list of trajectories with N being unknown beforehand. Each individual trajectory $\boldsymbol{y}_i$ consists of a set of 2D points, the time of appearance of its corner point into the trajectory, (i.e. the number of the image frame) and the optical flow vector of the last point in the list.

Initially, the list $\boldsymbol{Y}$ is empty. In every image frame, Harris corners are detected and the optical flow vector at each corner is estimated [5]. Then, each corner found in the current image frame is attributed to a trajectory that already exists

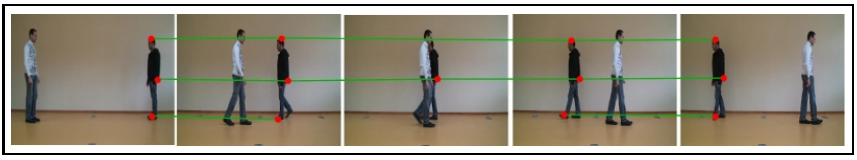


Fig. 1. Example of trajectories construction. The red dots represent the image key points and the green lines represent their trajectories. The figure is better seen in color.

or a new trajectory is created having with only one element, the corner under consideration. According to this scheme, three cases are possible:

- If any key point of the previous frame has an optical flow vector pointing out the key point under consideration, then the current corner is attributed to an existing trajectory. In this case, a trajectory follows the optical flow displacement vector, meaning that the corner is apparent in consecutive frames.
- If there is no such key point in the previous frame, we apply a window around the last corner which is similar to the current corner. If there are more than one similar corners then we choose the closet one.
- Otherwise, a new trajectory is constructed containing only the corner under consideration.

In Fig. 1, an intuitive example is presented where three corners are considered for demonstration purposes and five time instances are depicted. In the first frame, three corners are detected and three trajectories are created. In the second frame, the same corners are detected and associated with existing trajectories due to optical flow constraint. Next, one corner is detected and attributed to an existing trajectory due to optical flow constraints while the other two key points are occluded. During the fourth frame, the key point that was not occluded is also detected and attributed to an existing trajectory. One of the other two corner points that reappear is attributed to a trajectory due to local window matching. The other corner is not associated with any existing trajectory, so a new trajectory is created. In the last frame three corners are detected and associated with existing trajectories due to optical flow. Thus, four trajectories have been created, two of the same key point and another two of distinct key points.

Once the list $\boldsymbol{Y}$ is constructed trajectories of corner points belonging to the background are eliminated. This is achieved by removing the trajectories having small variance along the whole video sequence, according to a predefined-threshold value, as well as trajectories of small length (e.g. 1% of the number of frames). The complete procedure is described in the next algorithm 1.

3 Clustering Trajectories of Variable Length

Suppose we have a set of trajectories of N tracked feature points over T frames obtained from the previous procedure. The aim is to detect K independently

Algorithm 1. Trajectories construction algorithm

function CREATETRAJECTORIES($\boldsymbol{Im}$)
  Input: an image sequence $\boldsymbol{Im}$.
  Output: a list of trajectories $\boldsymbol{Y}$.
  $\boldsymbol{Y} = \emptyset$.
  for every image $\{im^{(t)}\}_{t=1,\ldots,T}$ **do**
    Detect corners $\{c_l^{(t)}\}_{l=1,\ldots,L^{(t)}}$ and estimate optical flow $\{f_l^{(t)}\}_{l=1,\ldots,L^{(t)}}$ in each one.
    for every corner $\{c_l^{(t)}\}_{l=1,\ldots,L^{(t)}}$ detected in $im^{(t)}$ **do**
      if $\boldsymbol{y}_i$ has its last corner c_i^{last} in the image $t-1$ and it's optical flow f_i^{last} points to the current corner, i.e. $c_i^{last} + f_i^{last} \approx c_l^{(t)}$ **then**
        Insert $c_l^{(t)}$ into $\boldsymbol{y}_i$.
      else if $\boldsymbol{y}_i$ has its the window around it's last corner c_i^{last} similar to the window around thw current corner $c_l^{(t)}$ **then**
        Insert $c_l^{(t)}$ into $\boldsymbol{y}_i$.
      else
        Insert a new trajectory $\boldsymbol{y}_i$ with only $c_l^{(t)}$ into $\boldsymbol{Y}$.
      end if
    end for
  end for
  Eliminate trajectory $\boldsymbol{y}_i$ with small variation in it's corners coordinates.
end function

moving objects in the scene by estimating labels on these points and classifying them into groups of different motions. Also, we want to estimate the characteristic motion of all objects.

A 2D trajectory $\boldsymbol{y}_i = (\boldsymbol{y}_i^{(1)}, \boldsymbol{y}_i^{(2)})$ consists of two directions: (1) horizontal and (2) vertical and is defined by a set of T_i points $\{(y_{i1}^{(1)}, y_{i1}^{(2)}), \ldots, (y_{iT_i}^{(1)}, y_{iT_i}^{(2)})\}$, corresponding to the successive image positions $(t_{i1}, \ldots, t_{iT_i})$ in the image sequence. That is important to note is that we deal with trajectories of variable length T_i, since occlusions or illumination changes may block the view of the objects in certain image frames.

Linear regression model constitutes a powerful platform for modeling sequential data that can be adopted in our case. In particular, we assume that a trajectory $\boldsymbol{y}_i^{(j)}$ of any direction $j = \{1, 2\}$ can be modeled through the following functional form:

$$\boldsymbol{y}_i^{(j)} = \boldsymbol{\Phi}_i \boldsymbol{w}^{(j)} + d_i^{(j)} + \boldsymbol{e}_i^{(j)} \ , \tag{1}$$

where $\boldsymbol{\Phi}_i$ is the design kernel matrix (common for both directions) of size $T_i \times T$, and $\boldsymbol{w} = (w_1, \ldots, w_T)$ is the vector of the T unknown regression coefficients that must be estimated. In our case we have considered a design matrix $\boldsymbol{\Phi}$ of size $T \times T$ having constructed with wavelets kernels. Thus, the matrix $\boldsymbol{\Phi}_i$ is a block-matrix of $\boldsymbol{\Phi}$, which has only the T_i lines that corresponds to the sucessive T_i frames $(t_{i1}, \ldots, t_{iT_i})$ of the i-th trajectory.

Also in the above equation, we assume an translation scalar term $d_i^{(j)}$ that allows for the entire trajectory to be translated as a unit [26]. Incorporating

of such term results in a regression model that allows for arbitrary translations in measurement space. Finally, the error term $\boldsymbol{e}_i^{(j)}$ in the above formulation is a T_i-dimensional vector that is assumed to be zero-mean Gaussian and independent over time, i.e. $e_i \sim \mathcal{N}(0, \Sigma_i)$ with a diagonal covariance matrix $\Sigma_i^{(j)} = diag(\sigma_{t_{i1}}^{2^{(j)}}, \ldots, \sigma_{t_{iT_i}}^{2^{(j)}})$[1].

Under these assumptions, the conditional density of trajectory is Gaussian, i.e. $p(\boldsymbol{y}_i^{(j)}|\boldsymbol{w}^{(j)}, \Sigma_i^{(j)}, d_i^{(j)}) = \mathcal{N}(\Phi_i \boldsymbol{w}^{(j)} + d_i^{(j)}, \Sigma_i^{(j)})$. In our case we consider the scalar $d_i^{(j)}$ to be a zero-mean Gaussian variable with a variance $u^{(j)}$, i.e. $p(d_i^{(j)}) = \mathcal{N}(0, u^{(j)})$. We can further integrate out d_i to obtain the marginal density for $\boldsymbol{y}_i^{(j)}$ which is also Gaussian,

$$p(\boldsymbol{y}_i^{(j)}|\theta) = \int p(\boldsymbol{y}_i^{(j)}|\boldsymbol{w}^{(j)}, \Sigma_i^{(j)}, d_i^{(j)}) p(d_i^{(j)}) dd_i^{(j)} = \mathcal{N}(\Phi_i \boldsymbol{w}^{(j)}, \Sigma_i^{(j)} + u^{(j)} \mathbb{1}) \ , \tag{2}$$

where $\mathbb{1}$ is a matrix of 1's of size $T_i \times T_i$. The marginal density is implicitly conditioned on the parameters $\boldsymbol{\theta} = \{\boldsymbol{w}, u, \Sigma\}$.

In our study we consider a different functional regression model for every object k, as described by the set of model parameters $\boldsymbol{\theta}_k = \{\boldsymbol{\theta}_k^{(1)}, \boldsymbol{\theta}_k^{(2)}\}$, where $\boldsymbol{\theta}_k^{(j)} = \{\boldsymbol{w}_k^{(j)}, u_k^{(j)}, \Sigma_k^{(j)}\}$. Therefore, the task of discovering K objects becomes equivalent to clustering the set of N trajectories into K clusters. This can be described by the following regression mixture models:

$$p(\boldsymbol{y}_i|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \pi_k p(\boldsymbol{y}_i|\boldsymbol{\theta}_k) = \sum_{k=1}^{K} \pi_k p(\boldsymbol{y}_i^{(1)}|\boldsymbol{\theta}_k^{(1)}) p(\boldsymbol{y}_i^{(2)}|\boldsymbol{\theta}_k^{(2)}) \ , \tag{3}$$

where we have assumed independence between that trajectories of both directions $(\boldsymbol{y}_i^{(1)}, \boldsymbol{y}_i^{(2)})$. In addition, π_k are the mixing weights satisfying the constraints $\pi_k \geq 0$ and $\sum_{k=1}^{K} \pi_k = 1$, while $\boldsymbol{\Theta}$ is the set of all the unknown mixture parameters, $\boldsymbol{\Theta} = \{\pi_k, \boldsymbol{\theta}_k\}_{k=1}^{K}$.

An important issue, when dealing with regression models is how to determine their order. Models of small order can lead to under-fitting, while larger order lead to curve over-fitting. Both cases may cause to serious deterioration of the clustering performance. Sparse modeling [27] offers a significant solution to this problem by employing models having initially many degrees of freedom than could uniquely be adapted given data. Sparse Bayesian regression can be achieved through a hierarchical prior definition over regression coefficients $\boldsymbol{w}_k^{(j)} = (w_{k1}^{(j)}, \ldots, w_{kT}^{(j)})^T$. In particular, we assume first that coefficients follows a zero-mean Gaussian distribution:

$$p(\boldsymbol{w}_k^{(j)}|\boldsymbol{\alpha}_k^{(j)}) = \mathcal{N}(\boldsymbol{w}_k^{(j)}|\mathbf{0}, \boldsymbol{A}_k^{-1^{(j)}}) = \prod_{l=1}^{T} \mathcal{N}(w_{kl}^{(j)}|0, \alpha_{kl}^{-1^{(j)}}) \tag{4}$$

[1] Again Σ_i is a blosck matrix of a $T \times T$ diagonal covariance matrix that corresponds to the noise variance of T frames.

where $\boldsymbol{A}_k^{(j)}$ is a diagonal matrix containing the T elements of precisions $\boldsymbol{\alpha}_k^{(j)} = (\alpha_{k1}^{(j)}, \ldots, \alpha_{kT}^{(j)})^T$. We impose next a Gamma prior on these hyperparameters::

$$p(\boldsymbol{\alpha}_k^{(j)}) = \prod_{l=1}^{T} Gamma(\alpha_{kl}^{(j)}|a,b) \propto \prod_{l=1}^{T} {\alpha_{kl}^{(j)}}^{a-1} \exp(-b\alpha_{kl}^{(j)}) \ , \tag{5}$$

where a and b denote parameters that are a prior set to near zero values (e.g. $a = b = 10^{-6}$). The above two-stage hierarchical prior is actually a Student-t distribution [27]. This is a heavy tailed prior distribution that enforces most of the values $\alpha_{kl}^{(j)}$ to be large, thus the corresponding $w_{kl}^{(j)}$ are set zero and eliminated from the model. In this way the complexity of the regression models is controlled in an automatic and elegant way and over-fitting is avoided.

Now the clustering procedure has been converted into a Maximum-A-Posterior (MAP) estimation approach, in a sense of estimating the mixture model parameters that maximize the MAP log-likelihood function:

$$L(\boldsymbol{\Theta}) = \sum_{i=1}^{N} \log\{\sum_{k=1}^{K} \pi_k p(\boldsymbol{y}_i|\boldsymbol{\theta}_k)\} + \sum_{k=1}^{K}\sum_{j=1}^{2}\{\log p(\boldsymbol{w}_k^{(j)}|\boldsymbol{\alpha}_k^{(j)}) + \log p(\boldsymbol{\alpha}_k^{(j)})\} \ . \tag{6}$$

In this direction, the Expectation - Maximization (EM) algorithm [28] can be applied in order to MAP estimate the model parameters $\boldsymbol{\Theta}$. It iteratively performs two main stages: During the *E-step* the expected values of the hidden variables are calculated. In our case this includes the cluster labels of trajectories as given by the posterior probabilities:

$$z_{ik} = P(k|\boldsymbol{y}_i) = \frac{\pi_k p(\boldsymbol{y}_i|\boldsymbol{\theta}_k)}{\sum_{k'} \pi_{k'} p(\boldsymbol{y}_i|\boldsymbol{\theta}_{k'})} \ , \tag{7}$$

as well as the mean value of the translations $d_{ik}^{(j)}$ at any direction. The latter is obtained by using the fact that the posterior density of translations is also Gaussian:

$$p(d_{ik}^{(j)}|\boldsymbol{y}_i^{(j)}, k) \propto p(\boldsymbol{y}_i^{(j)}|\boldsymbol{\theta}_k^{(j)}) p(d_{ik}^{(j)}) = \mathcal{N}(\hat{d}_{ik}^{(j)}, V_{ik}^{(j)}), \tag{8}$$

where

$$\hat{d}_{ik}^{(j)} = V_{ik}^{(j)} \left(\boldsymbol{y}_i^{(j)} - \Phi_i \boldsymbol{w}_k^{(j)}\right)^T {\boldsymbol{\Sigma}_{ik}^{-1}}^{(j)} \mathbf{1}_i \ \text{ and } \ V_{ik}^{(j)} = \left(\mathbf{1}_i^T {\boldsymbol{\Sigma}_{ik}^{-1}}^{(j)} \mathbf{1}_i + \frac{1}{u_k^{(j)}}\right)^{-1} , \tag{9}$$

where $\mathbf{1}_i$ is a T_i-length vector of 1's.

In the *M-step*, the maximization of the expected value of the complete log-likelihood function (Q-function) is performed. This leads to the following update rules [26], [29]:

$$\hat{\pi}_k = \frac{\sum_{i=1}^{N} z_{ik}}{N} \ , \tag{10}$$

$$\hat{\boldsymbol{w}}_k^{(j)} = \left[\sum_{i=1}^{N} z_{ik} \boldsymbol{\Phi}_i^T \boldsymbol{\Sigma}_{ik}^{-1^{(j)}} \boldsymbol{\Phi}_i + \boldsymbol{A}_k^{(j)} \right]^{-1} \sum_{i=1}^{N} z_{ik} \boldsymbol{\Phi}_i^T \boldsymbol{\Sigma}_{ik}^{-1^{(j)}} (\boldsymbol{y}_i^{(j)} - \hat{d}_{ik}^{(j)}) \ , \tag{11}$$

$$\alpha_{kl}^{(j)} = \frac{1 + 2a}{\hat{w}_{kl}^{2(j)} + 2b} \quad \forall \ l = 1, \ldots, T \ , \tag{12}$$

$$\hat{\sigma}_{kl}^{2^{(j)}} = \frac{\sum_{i=1}^{N} z_{ik} \left\{ \left(\boldsymbol{y}_{il}^{(j)} - [\boldsymbol{\Phi}_i \hat{\boldsymbol{w}}_k^{(j)}]_l - \hat{d}_{ik}^{(j)} \right)^2 + V_{ik}^{(j)} \right\}}{\sum_{i=1}^{N} z_{ik}} \ , \ \forall \ l = 1, \ldots, T \tag{13}$$

$$\hat{u}_k^{(j)} = \frac{\sum_{i=1}^{N} z_{ik} \left(\hat{d}_{ik}^{2(j)} + V_{ik}^{(j)} \right)}{\sum_{i=1}^{N} z_{ik}} \ . \tag{14}$$

where $[.]_l$ indicates the l-th component of the T_i-dimensional vector that corresponds to time location t_{il}.

After convergence of the EM algorithm, two kinds of information are available: At first the cluster labels of the trajectories are obtained according to the maximum posterior probability (Eq. 7). Moreover, the motion of objects are obtained from the predicted mean trajectories per cluster, i.e. $\boldsymbol{\mu}_k = (\boldsymbol{\mu}_k^{(1)}, \boldsymbol{\mu}_k^{(2)}) = (\boldsymbol{\Phi} \boldsymbol{w}_k^{(1)}, \boldsymbol{\Phi} \boldsymbol{w}_k^{(2)})$.

3.1 Initialization Strategy

A fundamental issue when applying the EM algorithm, is its strong dependence on the initialization of the model parameters due to its local nature. Improper initialization may lead to reaching poor local maxima of the log-likelihood, a fact that may affect significantly the performance of the method. A natural way for initialization is by randomly selecting K samples from the set of input trajectories, one for each cluster. Then, we can obtain the least-squared solution for the regression coefficients. In addition, the noise variance Σ_k is initialized by a small percentage of the total variance of all trajectories equally for each clusters, while we set the mixing weights π_k equal to $1/K$. Finally, one step of the EM algorithm is executed for further refining these parameters and calculating the MAP log-likelihood function. Several such different trials are made and the optimum solution is selected according to the likelihood function as the initial state of the parameters.

However, the above scheme cannot be easily applied to our approach due to the large variability in length (T_i) of the input trajectories which brings a practical difficulty in obtaining the least-squared solution. For this reason we have followed an alternative initialization scheme based on interpolation among successive time steps. More specifically, starting from the first time we perform periodically (e.g. every $0.05T$ frames) the k-means clustering over the points $(y_{it}^{(1)}, y_{it}^{(2)})$ until the

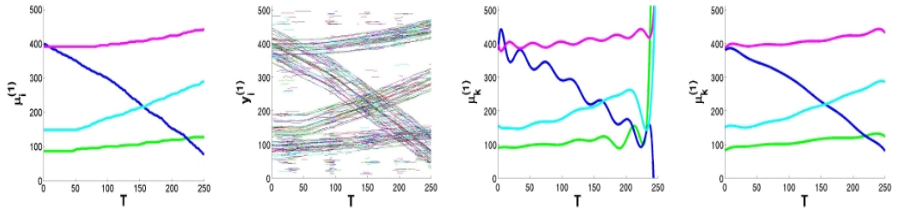


Fig. 2. The overall progress of our method to a experimental sequence of 250 images with $k = 4$ objects of different motion (a). First the input trajectories (b) are created, then an initial estimation of mean trajectories (c) are produced, and finally the predicted motion (d) is obtained

end of frame T. Then, the resulting K centers are associated with those of the previous time according to the minimum distance criterion. Finally, a linear interpolation (per cluster) is performed and thus the initial mean curves are produces used for estimating the initial values of the parameters. It must be noted that in cases where there is a large number of dense features representing the background, the initialization may diverge from the desired solution since the existence of a significant amount of outliers will affect the k-means solution. Even if during our experiments we have not faced with any such problem, treating this situation and eliminating this case still remains a future plan of study. An example of the proposed process is given in Fig. 2 adopted from a experimental data set, where both the initial interpolated trajectories and the final clustering solution are shown.

4 Experimental Results

We have studied the performance of our approach using both simulated and real examples. Some implementation details of our method are the following: At first, we have normalized spatial and temporal coordinates into the interval $[0, 1]$. Next, extracted trajectories with length less than 1% of the number of frames T were not taken into account. The same stands for those trajectories with variance less than 0.01. We have selected the mexican hat wavelet kernel $\phi(x) \propto \left(1 - \frac{x^2}{\sigma^2}\right) e^{\frac{-t^2}{2\sigma^2}}$. Experiments have shown that the method was not very sensitive to these kernel parameters. During our study we have set $\sigma = 0.3$. It must be noted that we have taken similar results for various values from the range $[0.1, 0.5]$.

Comparison has been made with the mean shift algorithm [1] which is a state of the art algorithm in visual tracking. For the mean shift algorithm, the images were represented in the RGB color space using histograms of 16 bins for each component. For initializing it we have manually selected the position and the size of each object in the first frame of the image sequence. After that, mean shift tracks the objects, using a distinct tracker for each target. The centers of

Fig. 3. Features are not uniformly distributed over the object and the center of gravity of the key points does not coincide with the center of gravity of the object. The small dots represent the features and the big dot represents their barycenter. The figure is better visualized in color.

the ellipses surrounding the targets are used to construct the mean trajectory of each object. This comparison favors mean shift in cases where the features are not uniformly distributed around the object, as the center estimated by the features may vary from the geometric center of the object (Fig. 3).

4.1 Experiments with Simulated Data Sets

The first series of experiments involves seven (7) simulated image sequences with spheres moving in predefined directions as shown in Fig. 4 and Fig. 5. Each sequence contains 130 frames of dimensions 512×512. About $1500 - 2000$ trajectories per problem were created with average length near 60 frames each. In sequences *Sim1* through *Sim5* all objects are visible during the whole sequence. In the next two experimental setups there is occlusion. In particular, in *Sim6* a sphere disappears while in *Sim7* a sphere disappears and reappears.

Since in our study we are aware of the ground truth, the performance of both tracking approaches was evaluated using two criteria:

- The mean squared error (MSE), measured in pixels, between the ground truth $\boldsymbol{r}$ and the estimated mean trajectories $\boldsymbol{\mu}$ as given by

$$MSE = \frac{1}{K \cdot T} \sum_{k=1}^{K} \sum_{j=1}^{2} ||r_k^{(j)} - \mu_k^{(j)}||^2 \ .$$

- The percentage of correctly classified trajectories (ACC). It must be noted that the input trajectories created by our method have been chosen also to evaluate mean shift algorithm. In particular, we assign every input trajectory to an object according to the smallest distance with the predicted mean trajectory.

Table 1 shows the results obtained by the proposed method and the mean shift algorithm. As it may be seen, both methods have comparable accuracy. However, our approach estimates better the true motion of the objects, as shown from the MSE criterion. Also in problem *Sim7*, mean shift fails to track the sphere that

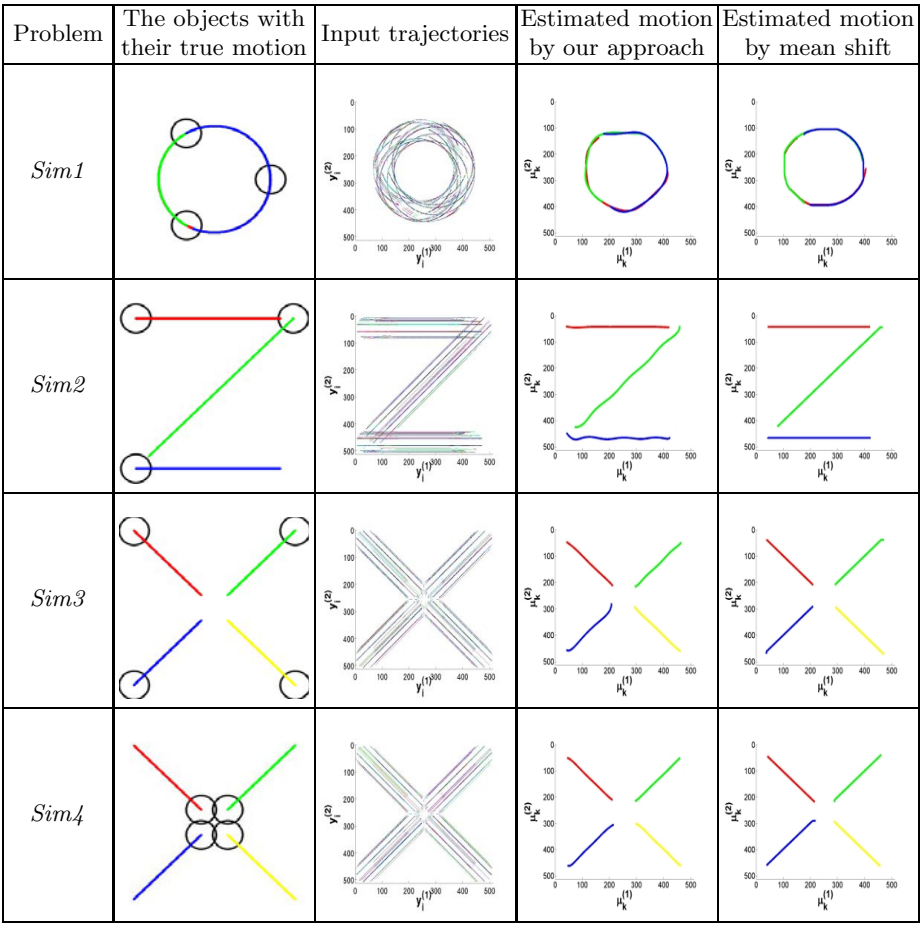


Fig. 4. A part of the simulated data sets used in our experimental study. For each problem we give the real objects motion, the created input trajectories and the predicted motion as estimated by both compared approaches.

disappears and reappears and tracks the object only as long as it is visible. In the case of problems *Sim6* and *Sim7*, the frames in which a sphere is not visible are not taken into account for computing MSE. On the other hand, the proposed method correctly associates the two separated trajectories of the sphere.

4.2 Experiments with Real Data Sets

We have also studied our motion segmentation approach on five (5) real sequences shown in Fig. 6. Three of them (*Real1-3*) show mobile robots moving in various directions and two other sequences (*Real4-5*) contain two persons walking. In there, occlusions take place, as one person gets behind the other. All images are of size 512×512 pixels. During the set *Real1* ($T = 250$), the robots

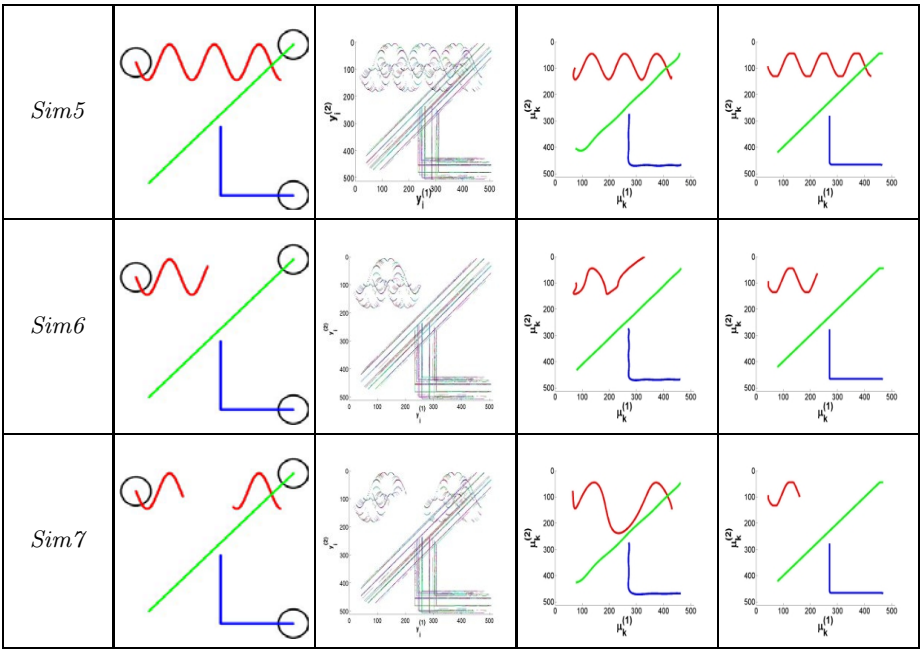


Fig. 5. A part of the simulated data sets used in our experimental study. For each problem we give the real objects motion, the created input trajectories and the predicted motion as estimated by both compared approaches.

Table 1. The performance of our approach and mean shift in terms of classification accuracy and mean squared error criteria

Problem	Our approach		Mean shift	
	MSE	ACC	MSE	ACC
Sim1	69	100%	121	100%
Sim2	10	99%	114	100%
Sim3	10	96%	114	99%
Sim4	15	97%	130	99%
Sim5	20	100%	118	100%
Sim6	29	100%	74	100%
Sim7	41	99%	lost	lost

are moving towards the borders of the image forming the vertices of a square. In *Real2* ($T = 680$), the robots are moving around the center of the image, forming a circle, while in *Real3* ($T = 500$), the robots are moving forward and backward. Finally, in *Real4* ($T = 485$) two persons are moving from the one side of the scene to the other and backwards, and in *Real5* ($T = 635$) the persons are not only moving from one side to the other many times but also they move forward and backward in the scene. In problems *Real4* and *Real5*, due to occlusions,

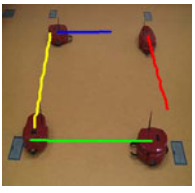
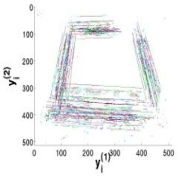
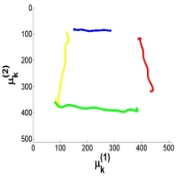
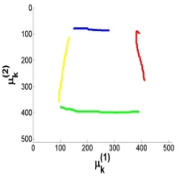
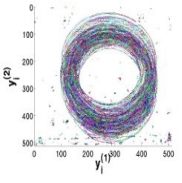
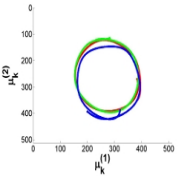
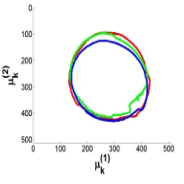
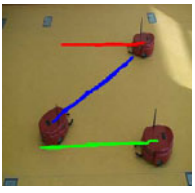
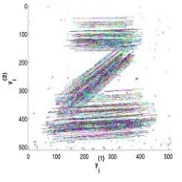
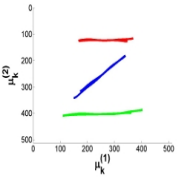
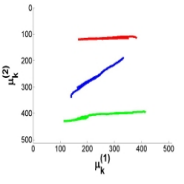
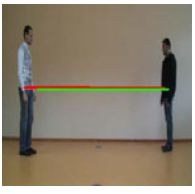
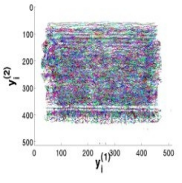
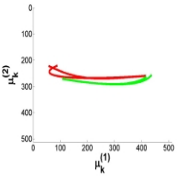
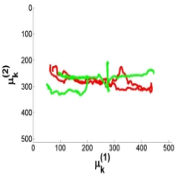
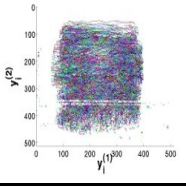
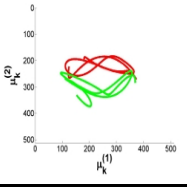
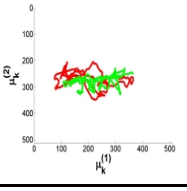
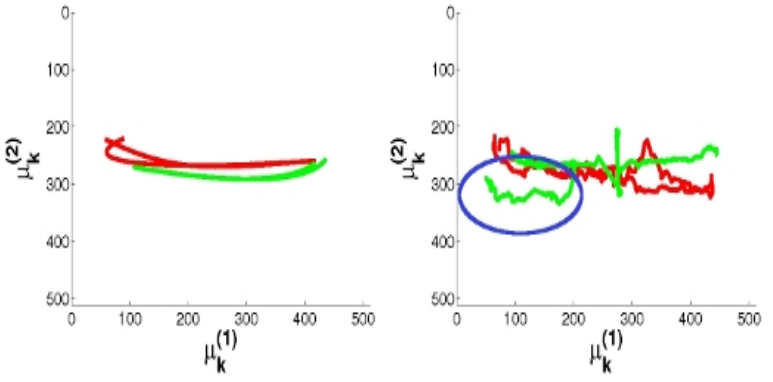
Problem	The objects with their true motion	Input trajectories	Estimated motion by our approach	Estimated motion by mean shift
Real1				
Real2				
Real3				
Real4				
Real5				

Fig. 6. The five real data set used in our experimental study. For each problem we give the real objects motion (chosen manually), the created input trajectories and the predicted motion as estimated by both comparative approaches.

the mean shift algorithm fails to track the objects while the proposed algorithm successfully follows them.

As ground truth is not provided for these sequences, only visual evaluation can be done. In problems *Real1-3* both algorithms produce approximately the same trajectories. On the contrary, in problems *Real4* and *Real5*, where we deal



(a) (b)

Fig. 7. Estimated trajectories for the sequence *Real4*. (a) Our method, (b) mean shift. The green (printed in light gray in black and white) trajectory in (b) corresponds to the person in black moving from the right side of the image to the left and backwards. The ellipse highlights the part of the trajectory where the person is lost, because mean shift fails to track the object due to occlusion. The figure is better visualized in color.

with articulated objects and occlusion, mean shift does not produce smooth trajectories and one object is lost. This is due to the change in the appearance of the target. When the person walks there are instances where both arms and legs are visible and instances where only one of them is present. In these cases, mean shift identifies the target with respect to its initial model and may produce abrupt changes in motion estimation. On the other hand, our method smooth out these effects through data association.

Looking in more detail the problem *Real4*, we can see the person in black disappears (because he gets behind the other person) twice during this sequence: at first, when he is moving from right to left, and then, as he is moving from left to right. Mean shift successfully follows the object before and after the first occlusion, but it fails to track it then the second one takes place. This is better depicted in Fig. 7(b) where the predicted trajectory, corresponding to the person in black, is shown in green. The part of the trajectory where the object is lost is highlighted by an ellipse. On the other hand, the proposed method successfully tracks the object in all frames (Fig. 7(a)).

5 Conclusion

We have presented a compact methodology for objects tracking based on model-based clustering trajectories of Harris corners extracted from a video sequence. Clustering is achieved through an efficient sparse regression mixture model that embodies special characteristics in order to handle trajectories of variable length, and to be translated in measurement space. Experiments have shown the abilities of our approach to automatically detect the motion of objects without any human

interaction and also have demonstrated its robustness to occlusion and feature misdetection. Some directions for future study include an alternative strategy for initializing mixture model parameters during EM procedure especially in cases of occlusion, as well as to simultaneously estimate the number of objects K in the image sequences. Also, our willing is to study the performance of our method in other interesting computer vision applications, such as human action recognition [30], as well as to fully 3D motion estimation.

References

1. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(5), 564–577 (2003)
2. Cuevas, E., Zaldivar, D., Rojas, R.: Kalman filter for vision tracking. Technical Report B 05-12, Freier Universitat Berlin, Institut fur Informatik (2005)
3. Yang, M., Wu, Y., Hua, G.: Context-aware visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(7), 1195–1209 (2009)
4. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. International Journal of Computer Vision 29, 5–28 (1998)
5. Shi, J., Tomasi, C.: Good features to track. In: 1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1994), pp. 593–600 (1994)
6. Wang, Z., Yang, X., Xu, Y., Yu, S.: Camshift guided particle filter for visual tracking. Pattern Recognition Letters 30(4), 407–413 (2009)
7. Zhoo, H., Yuan, Y., Shi, C.: Object tracking using SIFT features and mean shift. Computer Vision and Image Understanding 113(3), 345–352 (2009)
8. Zhao, Q., Tao, H.: Differential Earth Mover's Distance with its application to visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(5), 274–287 (2010)
9. Karavasilis, V., Nikou, C., Likas, A.: Visual tracking using the earth mover's distance between Gaussian mixtures and Kalman filtering. Image and Vision Computing 29(5), 295–305 (2011)
10. Black, M., Anandan, P.: The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. Computer Vision and Image Understanding 63(1), 75–104 (1996)
11. Wong, K.Y., Spetsakis, M.E.: Motion segmentation by EM clustering of good features. In: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW 2004), vol. 11, pp. 1–8. IEEE Computer Society, Los Alamitos (2004)
12. Pundlik, S.J., Birchfield, S.T.: Real-time motion segmentation of sparse feature points at any speed. IEEE Transactions on Systems, Man, and Cybernetics 38, 731–742 (2008)
13. Buzan, D., Sclaroff, S., Kollios, G.: Extraction and clustering of motion trajectories in video. In: International Conference on Pattern Recognition, pp. 521–524 (2004)
14. Alon, J., Sclaroff, S., Kollios, G., Pavlovic, V.: Discovering clusters in motion time-series data. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 375–381 (2003)
15. Zappella, L., Llado, X., Provenzi, E., Salvi, J.: Enhanced local subspace affinity for feature-based motion segmentation. Pattern Recognition 44, 454–470 (2011)

16. Zeppelzauer, M., Zaharieva, M., Mitrovic, D., Breiteneder, C.: A novel trajectory clustering approach for motion segmentation. In: Boll, S., Tian, Q., Zhang, L., Zhang, Z., Chen, Y.-P.P. (eds.) MMM 2010. LNCS, vol. 5916, pp. 433–443. Springer, Heidelberg (2010)
17. Lauer, F., Schnorr, C.: Spectral clustering of linear subspaces for motion segmentation. In: Proc. of the 12th IEEE Int. Conf. on Computer Vision, ICCV 2009 (2009)
18. Julià, C., Sappa, A., Lumbreras, F., Serrat, J., López, A.: Motion segmentation from feature trajectories with missing data. In: Martí, J., Benedí, J.M., Mendonça, A.M., Serrat, J. (eds.) IbPRIA 2007. LNCS, vol. 4477, pp. 483–490. Springer, Heidelberg (2007)
19. Yan, J., Pollefeys, M.: A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 94–106. Springer, Heidelberg (2006)
20. Kim, J.H., Agapito, L.: Motion segmentation using the Hadamard product and spectral clustering. In: Proceedings of the 2009 International Conference on Motion and Video Computing, WMVC 2009, pp. 126–133. IEEE Computer Society, Los Alamitos (2009)
21. Masoud, O., Papanikolopoulos, N.P.: A novel method for tracking and counting pedestrians in real-time using a single camera. IEEE Transactions on Vehicular Technology 50(5), 1267–1278 (2001)
22. Harris, C., Stephens, M.: A combined corner and edge detection. In: Proceedings of The Fourth Alvey Vision Conference, pp. 147–151 (1988)
23. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI 1981), pp. 674–679 (1981)
24. Lowe, D.G.: Distinctive image features from Scale-Invariant keypoint. International Journal of Computer Vision 60(2), 91–110 (2004)
25. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. International Journal of Computer Vision 65(1/2), 43–72 (2005)
26. Gaffney, S.: Probabilistic curve-aligned clustering and prediction with regression mixture models. PhD thesis, Department of Computer Science, University of California, Irvine (2004)
27. Tipping, M.: Sparse Bayesian Learning and the Relevance Vector Machine. Journal of Machine Learning Research 1, 211–244 (2001)
28. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. J. Roy. Statist. Soc. B 39, 1–38 (1977)
29. Blekas, K., Nikou, C., Galatsanos, N., Tsekos, N.: A regression mixture model with spatial constraints for clustering spatiotemporal data. Intern. Journal on Artificial Intelligence Tools 17(5), 1023–1041 (2008)
30. Oikonomopoulos, A., Patras, I., Pantic, M., Paragios, N.: Trajectory-based representation of human actions. In: Huang, T.S., Nijholt, A., Pantic, M., Pentland, A. (eds.) ICMI/IJCAI Workshops 2007. LNCS (LNAI), vol. 4451, pp. 133–154. Springer, Heidelberg (2007)

PTMSearch: A Greedy Tree Traversal Algorithm for Finding Protein Post-Translational Modifications in Tandem Mass Spectra

Attila Kertész-Farkas[1], Beáta Reiz[2,3], Michael P. Myers[1], and Sándor Pongor[1,2]

[1] International Centre for Genetic Engineering and Biotechnology, Padriciano 99, I-34012 Trieste, Italy
[2] Bioinformatics Group, Biological Research Centre, Hungarian Academy of Sciences, Temesvári krt. 62, H-6701 Szeged, Hungary
[3] Institute of Informatics, University of Szeged, Aradivértanúk tere 1, H-6720, Szeged, Hungary
kfattila@icgeb.org

Abstract. Peptide identification by tandem mass spectrometry (MS/MS) and database searching is becoming the standard high-throughput technology in many areas of the life sciences. The analysis of post-translational modifications (PTMs) is a major source of complications in this area, which calls for efficient computational approaches. In this paper we describe PTMSearch, a novel algorithm in which the PTM search space is represented by a tree structure, and a greedy traversal algorithm is used to identify a path within the tree that corresponds to the PTMs that best fit the input data. Tests on simulated and real (experimental) PTMs show that the algorithm performs well in terms of speed and accuracy. Estimates are given for the error caused by the greedy heuristics, for the size of the search space and a scheme is presented for the calculation of statistical significance.

1 Introduction

Tandem mass spectrometry (MS/MS) has become the major tool for high throughput protein analysis in the biomedical field, and the analysis of the data entirely relies on database searching algorithms. The difficulties in analysis arise from both the quantity of data, as current instruments can produce tens of thousands of spectra within an hour, and the quality of data, as there are measurement errors, as well as both missing and extraneous data points.

According to the most widely used methodology, peptide sequences, taken from a sequence database, are matched to the experimental MS/MS spectra in order to suggest peptide candidates for further analysis. The algorithmic strategy is seemingly straightforward: protein sequences are divided into peptide segments that in turn are converted into theoretical spectra according to chemical rules and stored in a database [1]. Each experimental MS/MS spectrum is then compared with theoretical spectrum from the database, and the most similar peptide is stored for further analysis.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 162–176, 2011.

The difficulties come from the fact that experimental and theoretical spectra can differ for reasons other than instrument noise. One of the most important sources of variability are the post-translational modifications (PTMs) of proteins. In biological systems, proteins can carry a number of PTMs on their amino acid side-chains which leads to a combinatorial explosion in the number of theoretical spectra to be compared. As there are thousands of already discovered PTMs [2], accurate identification of PTMs through analysis of high-throughput MS/MS data is considered a highly challenging problem [3], and this is the subject of our work.

Here we present PTMSearch, a greedy-tree search algorithm designed for the identification of PTMs in tandem mass spectrometry data. The approach is based on a tree-representation of the search space in which nodes are amino acid positions and branches represent potential PTMs allowed on the type of amino acid in the given position. In this representation, a path from the root to a leaf represents a peptide with PTMs. PTMSearch uses a greedy traversal procedure to find the path in the tree that best fits to the input data. PTMSearch includes tree pruning heuristics in order to keep the computation time reasonable and polynomial.

The rest of this paper is structured as follows: Section 2 is a brief outline of the spectrum comparison problem and the role of PTMs in the process (optional). Section 3 is a summary of related work. In Section 4 we describe our proposed algorithm and the speedup methodologies. Section 5 describes the calculation of significance used with the algorithm. Section 6 contains the results obtained on experimental datasets and comparisons with other method. Section 7 is a discussion of the results and an outline of future work.

2 MS/MS Spectra and PTMs

This section presents a simplified description of mass spectra for readers not familiar with the field. Of the various and continuously changing techniques of mass spectrometry, we selected the most widely used method, collision induced dissociation (CID), to demonstrate the principle of spectrum analysis. Tandem mass spectra of peptides (MS/MS spectra, hereafter: spectra) are produced by mass spectrometers by fragmenting peptides (typically up to 8-30 amino acids in length) at given positions of the peptide chain. In theory, fragmentation results in a series of peptide fragments of which two types, the b and y ions are used most frequently for identifying peptides. A theoretical fragmentation pattern can be calculated based on the sequence alone, this is called the theoretical spectrum, the x axis is the molecular mass of the fragment peak (more exactly the mass over charge ratio m/z of the fragment ion) the y axis is proportional to the quantity of the ions. Since predicting the intensity *in silico* has been proven difficult, the intensity in the theoretical spectra are set to unit. If a post-translational modification (PTM) occurs at one position (amino acid) of the peptide (say at the M residue), than the (m/z) of all fragments in which this position is included will be shifted by a value corresponding to the modification. This will result in an altered theoretical spectrum [4].

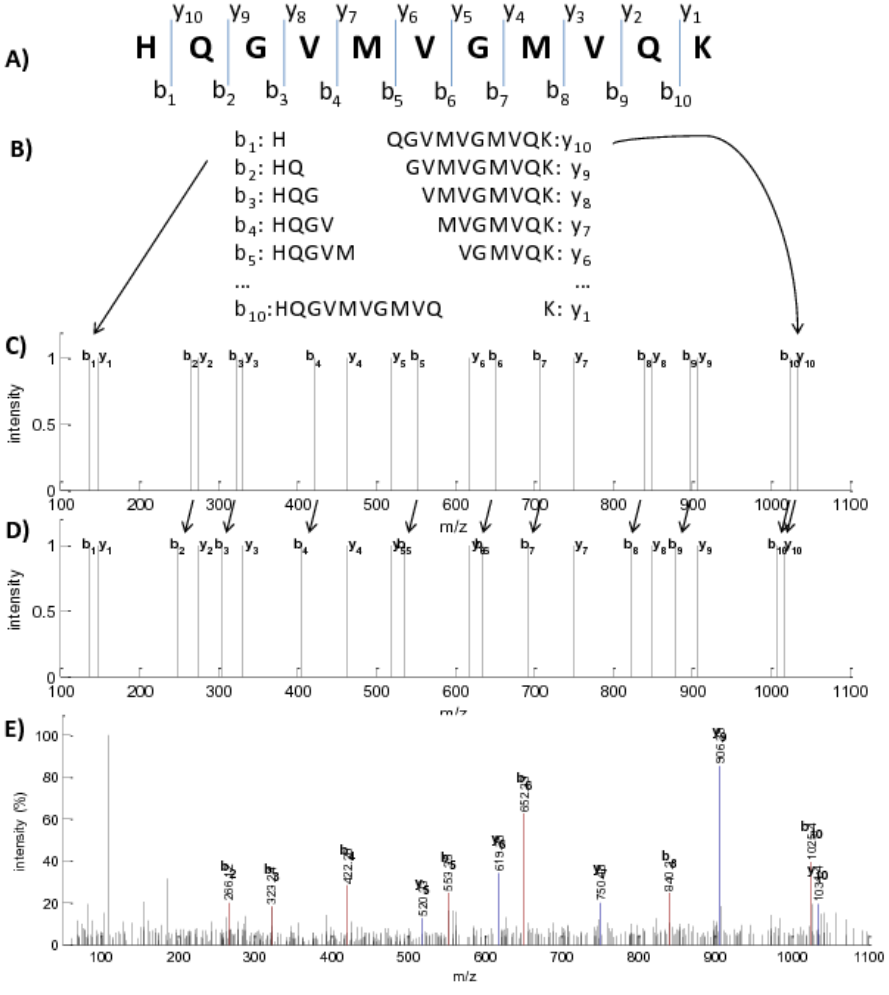


Fig. 1. A simplified outline of peptide fragmentation, shown on the example peptide HQGVMVGMVQK A) In the mass spectrometer, peptides are fragmented at certain points of the backbone. B) In theory, the resulting b and y ions form a regular series. C) This series is represented as a theoretical spectrum. D) If a modification is introduced, say at the 2nd residue (Q) of the sequence, all b and y ions that contain that residue will be shifted by a value corresponding to the molecular mass of the modification (see arrows). In this example we show the effect of deamidation at Q in position 2, which decreases the molecular mass by 17.03 Daltons (small negative shift indicated by arrows). E) An experimental spectrum of the same peptide recorded by a CID mass spectrometer. The annotated peaks correspond to the $b-y$ fragment ions. Some $b-y$ ions e.g. b_1, y_9 are not observed in this example, all other peaks can be considered noise.

Eventhough experimental spectra produced by the mass spectrometer may contain only some of the peaks predicted by the theoretical spectra, the peak intensities (y axis) vary in a broad range and the spectrum is always laden with noise peaks, a classical numerical similarity measure should be able to tell if the experimental spectrum is closer to the modified or to the unmodified spectrum, even if the overlaps of noise peaks with modified peaks can obscure the results.

One of the simplest similarity measure for two spectra (an experimental and a theoretical spectrum) is the so called shared peak count (SPC), which is the number of peaks that are on the same position (m/z) within a given tolerance range, called (fragment) ion tolerance. Formally, for an experimental spectrum q and theoretical spectrum t, the number of shared peaks can be computed as,

$$SPC(q,t) = |\{(q_i, t_j) : |q_i - t_j| < \delta\}|, \tag{1}$$

where a_i is the location (m/z) of ith in the spectrum a and δ is the ion tolerance. In the case of PTMs, the algorithms must take into account the mass difference of the precursor ion and the resulting fragment ions, as the algorithms typically compare experimental and theoretical spectra coming from precursors with similar masses. Since we do not know which PTM to expect, we need to produce theoretical spectra for all of them. There are many different kinds of protein modifications (the Unimod database lists over 500 types of them `www.unimod.org`) and in most cases the modifications are only partial i.e. both the modified and the unmodified amino acid can be present. This results in a combinatorial explosion, which makes the accurate identification of PTMs an especially challenging problem. For instance, if we allow 5 of the 10 most frequent modifications to occur in a peptide, the search space grows 3 orders of magnitude, but the growth is more dramatic if instead of 10 types of modifications we wish to consider all of roughly 500 known types [5].

3 Related Work

There is a large body of literature on the computational identification of PTMs in MS/MS spectra (for excellent reviews see [6]). The methods fall in 3 large categories:

- Targeted PTM identification. In this case, the user/experimenter has to define the types of PTMs and input these as parameters to the program.
- Untargeted PTM identification. This approach uses a full list of known modifications as a vocabulary, such as UniMod (`www.unimod.org`) or ResId (`http://www.ebi.ac.uk/RESID/`).
- Unrestricted PTM identification. (also called de novo or blind PTMs identification). Here, no assumptions are made about the PTMs, but all shifts that obey the chemical rules (and are recurrent in a dataset) are considered as potential PTMs. Here, novel PTMs can be identified, but unfortunately the statistical significance of the hits are often low, especially when more then one PTM per peptide is allowed.

In practice, search engines running on single CPU computers can handle up to 4-5 modifications per peptide, above this limit the search space becomes too large and analysis often becomes too time-consuming [5]. From the many programs available we mention two:

MS-Alignment. The Pevzner group has proposed a model for unrestrictive PTM identification that seeks to identify an optimal alignment between the experimental and the theoretical spectrum that maximizes the overlap of the peaks between the two spectra via peak shifting [7]. One shift represents one PTM and the size of the shift will represent the molecular mass of the PTM. The method is analogous to finding an optimal edit distance between two strings. The program is freely available and an online web server can be found at: `http://proteomics.ucsd.edu/LiveSearch/`.

PILOT_PTM.(Prediction via Integer Linear Optimization and Tandem mass spectrometry) uses a binary integer optimization model for finding PTMs that best match the experimental data [8]. This is an untargeted method i.e. PILOT_PTM uses a vocabulary of PTMs. Given a template peptide sequence of amino acids, the model will seek to determine the optimal set of modifications among a "universal" list of PTMs. Then the obtained binary linear model is solved by relaxing the variables and cutting plane techniques.

We describe an alternative approach in which the PTM search space is mapped to a tree structure and finding an optimal set of PTMs is reduced to a tree traversal problem. The resulting tree is still too big for the classical, exhaustive tree traversal algorithms (such as depth-first search, breadth-first search [9]), so we developed a greedy traversal algorithm.

4 Method: The PTMSearch Algorithm

The PTMSearch algorithm compares an experimental spectrum to a peptide sequence in order to identify PTMs untargeted way. Briefly, the algorithm generates all possible modifications of a peptide sequence and selects the set of modifications for which the precursor mass of the modified peptide is within a small tolerance of the precursor mass of the spectrum. If there is more than one such set of PTMs, the algorithm selects the one that shares the maximum number of peaks with the experimental spectrum (eqn 1). Since the search space is prohibitively large, PTMSearch uses a greedy approach and speedup techniques in finding the optimum.

Let q be an experimental spectrum and $p = a_1 a_2 \dots a_n$ be a peptide sequence and let's denote the precursor mass of q and p by $PM(q), PM(p)$ respectively. Let L_a be a list of modification masses for the amino acid a and $n_a = |L_a|$ denote the number of the elements in the list. A modification mass is the mass difference that the amino acid gains or looses due to the molecular modification. For example $L_c = \{-17.0265, 47.9847, 57.0215, 71.0371, \dots\}$ means the amino acid cysteine can loose 17.065Da (occurs via losing of ammonia from cysteine), can gain 47.9847Da (occurs via complete oxidation), can gain 57.0215Da (occurs via carbamidomethylation), etc. respectively. The molecular mass of the cysteine

is 121.16Da which becomes 178.1815 after carbamidomethylation. Note that one particular amino acid molecule is not modified with more than one modification at the same time, but an amino acid can be modified with various modification at different occurrences in the peptide sequence (and in different peptides as well).

The basic idea is to generate all modified variations of peptide p and store them in a prefix tree, where a branch at the level i denotes PTM on amino acid a_{i+1}.

More formally, the tree node at the level i is a structure $v = \langle s, b, y, m, c\rangle$, where s is a score of the node, c is the number and m is the sum of the mass of the acquired modifications in the sequence $a_1 \dots a_i$. Finally the variables b and y store the masses (m/z) of the b and y fragment ions that correspond to the $a_1 \dots a_i$ and $a_{i+1} \dots a_n$ fragment ions respectively.

We recursively define the tree and the values stored in the node structures as follows: The root node is $\langle 0, 0, PM(q), 0, 0\rangle$. If $v = \langle s, b, y, m, c\rangle$ is a node in the tree at the level $i \quad (0 \leq i < n)$ then the node $v_0 = \langle s', b + m_{a_{i+1}}, y - m_{a_{i+1}}, m, c\rangle$ is a child of the node v at level $i + 1$, $m_{a_{i+1}}$ is the mass of the amino acid a_{i+1}, and $s' = s + \widetilde{s}(b') + \widetilde{s}(y')$, where

$$\widetilde{s}(x) = \begin{cases} 1 & \text{if } \exists j : |q_j - x| < \delta \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where q_j denotes the (m/z) location of the jth peak in the spectrum q and δ is the ion match tolerance. The nodes $v_j = \langle s', b + m_{a_{i+1}} + m_j, y - m_{a_{i+1}} - m_j, m + m_j, c{+}1\rangle$ are children of the node v at level $i{+}1$, where $j = 1, \dots, n_{a_{i+1}}$, m_j is the mass of the jth modification in $L_{a_{i+1}}$. s' is defined the same as earlier. The node v_j at level $i + 1$ represents that the amino acid a_{i+1} in the peptide p is modified with the jth modification from $L_{a_{i+1}}$. If the node v is at the level n, then node v is a leaf. Denote $w(v) = s$ as the score of the node v. A leaf $\langle s, b, y, m, c\rangle$ is called a goal leaf if $|PM(q) - PM(p)| = m$ up to a small precursor mass tolerance, that is the mass of the peptide with the acquired modification masses is equal to the precursor mass of the query spectrum.

We denote this computation tree T and the subtree of T rooted at a node v is denoted T_v. The number of the nodes in the tree T_v is denoted by $|T_v|$.

The score of the spectrum q is the maximum of the scores of the goal leaves, denoted $H(q) = \max\{w(v) \mid v \text{ is a goal leaf in tree } T\}$. This goal can be found with any kind of tree traversal methods like Depth-first, Breadth-first traversal algorithms. The modifications on the peptide then can be extracted from the path between the root and the best goal leaf.

Note that, all nodes at level $i \quad (0 \leq i < n)$ correspond to the ith amino acids in p, all have the same $n_{a_i} + 1$ children respectively, and the tree is balanced and all leaves have the same depth. Thus, the number of the nodes in the tree is $|T| = \prod_{i=1}^{n}(n_{a_i} + 1)$, which makes the time complexity of the traversal algorithm impractical. Note that the size of the search space does not depend on the experimental spectrum q. In the next subchapter we define pruning techniques in order to maintain the run time polynomial and appropriate for real applications.

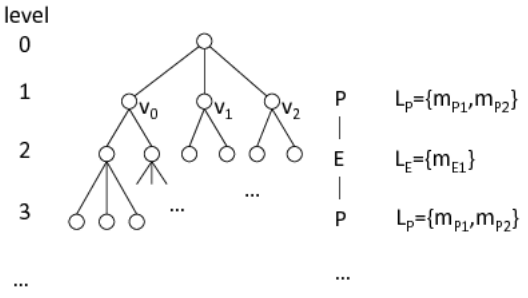


Fig. 2. Sketch of the computation tree for a peptide. PEP stands for the peptide sequence starting from above. The nodes at the level i represent the $a_1 \dots a_i$ peptide fragments modified in all possible ways.

4.1 Speedup Techniques

In this section we present various speedup solutions that prune the search space and make the algorithm suitable for real-life applications.

Limiting the number of PTMs. If we restrict the number of the PTMs on a peptide to K, then a node $\langle s, b, y, m, c \rangle$ can be deleted from the tree if $c > K$. Thus, the number of the nodes $|T'|$ in the pruned tree T' can be estimated by the following formula

$$m^K \binom{n}{K} \leq |T'| \leq M^K \binom{n}{K} \leq \underbrace{\frac{M^K}{K!}}_{\text{const.}} n^K, \tag{3}$$

where $m = \min_a\{n_a\}$ and $M = \max_a\{n_a\}$, which gives that the number of the nodes in the tree T' is $O(n^K)$.

Figure 3 plots a distribution of the number of modifications per amino acid in our PTM database. See more details about PTM dataset we used in the Experimental results chapter. There is one amino acid that can be modified by 32 different PTMs, so $M = 32$ and there are 5 amino acids that cannot be tagged by any modification, so $m = 0$. We can say loosely that the expected number N_L of modifications per amino acid is around 5 or 6.

Greedy Tree Traversal. In principle the tree can be traversed by any of the known tree traversal algorithms. Here, we present a Greedy Tree Traversal (GTT) algorithm that inserts a node to a queue Q when it is visited at the first time, deletes when all of its children have been visited, and continues the search from the node with the highest score in the Q. When the size of Q exceeds a certain limit Q_T, the node with the lowest score is deleted along with the corresponding subtree.

The queue Q can be implemented as a priority double ended queue, where the ordering operator of elements is based on the relation "$<$" over the score of

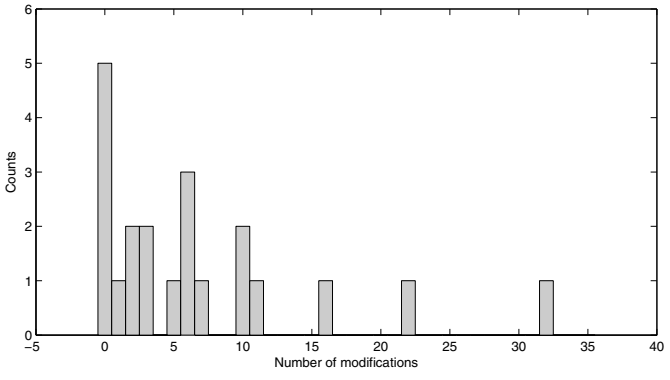


Fig. 3. Distribution of the number of PTMs per amino acids in our PTM database

the nodes. In this case, a new node is inserted at the end of the nodes with equal scores in Q. This algorithm can be described best by the following pseudocode.

Algorithm 1. Greedy Tree Traversal (GTT)
input: Tree T
output: best goal (or NULL if there is no goal leaf)
1 put(Q,root);
2 **while** Q is not empty
3 v = front(Q);
4 create a non-visited child v_j of node v;
5 **if** all children of v has been visited **then** pop_first(Q);
6 **if** v_j is a goal leaf **then** update best goal;
7 **else** put(Q,v_j);
8 **if** size(Q) $> Q_T$ **then** pop_last(Q);
9 **return** best goal (or NULL if there is no goal leaf);

It may happen that the true goal leaf is eliminated from the tree by deleting a node in the 8th code line. Now, we give an estimate about the probability of eliminating the true goal under the following assumptions: The probability of a node v matches to a peak by chance (i.e. the score of a child of v is increased due to one of the fragment ions match with a noise peak) is $p = 2 * m * \delta / PM(q)$, where m is the number of the peaks in q. We assume that the peaks are evenly and independently distributed in the experimental spectrum. Let p_e be the probability of that a (non-noise) peak is missing from the spectrum independently from other peaks (because either it was not observed or was filtered out in a preprocessing step).

Theorem 1. *Using the assumptions and parameters above the probability* $P(\epsilon)$ *of eliminating the true goal from the search space is* $P(\epsilon) = N_L \cdot p(K + \sum_{j=1}^{H} p_e^j)$, *where* $H = Q_T/(N_L)^K$, $Q_T > M$ *is the bound of the size of the queue* Q, *and* K *is the limit of the PTMs on a peptide to be identified.*

Proof. First case: Let's assume the GTT is on the right path to the true goal at node v at level i, and the b_{i+1} and y_{n-i+1} peaks are not missing. GTT can leave the true path iff one of the children l of the node v hits a random match by chance before the right child is extended. In this case GTT will visit all nodes of the T_l because all nodes in T_l have greater weight than any other nodes in Q. If $|T_l| > Q_T$, then the node v will be deleted from the Q and the right path to the best node will be lost. Since the path from the root to the true goal has K branches, each has N_L branches, then the probability of this error $p_{\epsilon_1} = K \cdot N_L \cdot p$.

Second case: Let's assume GTT is on the right path to the true goal at node v at level i, v has a match, and the b_{i+1} and y_{n-i+1} peaks are missing. v has the unique and greatest score $w(v)$ in Q. ($w(v)$ must be unique, since v had a match and v's parent has smaller weight. If there is another node x such that $w(x) = w(v)$ in the Q, then GTT first traversals T_x, deletes x from Q, then returns back to T_v.) GTT starts visiting the nodes in T_v Breadth-first way until a node matches to a peak by chance. Q can store Q_T nodes of T_v, $H = Q_T/N_L^K$ levels. So, H right peaks must be missing from the spectra successively, before GTT starts deleting nodes of T_v from Q. The probability of that H consecutive peaks are missing is p_e^H. Thus, the probability of error of the second case is $p_{\epsilon_2} = p \cdot N_L \cdot \sum_{j=1}^{H} p_e^j$.

Then, the final probability of eliminating the true goal from the search space can be obtained by summing the cases since they can happen independently. Thus, we have

$$P(\epsilon) = p_{\epsilon_1} + p_{\epsilon_2} = K \cdot N_L \cdot p + p \cdot N_L \cdot \sum_{j=1}^{H} p_e^j = N_L \cdot p(K + \sum_{j=1}^{H} p_e^j). \quad (4)$$

□

Consequence 1. Deleting a node from the tree polynomially reduces the search space. If we decrease the Q_T threshold, more nodes will be eliminated however the error will grow only linearly.

Consequence 2. The probability $P(\epsilon)$ can lessen with more accurate resolution of the spectra, (smaller δ).

Consequence 3. The probability $P(\epsilon)$ can lessen via removing more noise peaks from the experimental data in order to decrease the chance of random matches.

Remark 1. This greedy approach can be considered as a special case of the $A^\star$ algorithm [9], where the distance-plus-cost function is defined $h(n) = g(n) + f(n)$ for a node n, where $g(n) = w(n)$ is the score from the root to the node n and $f(n)$ is an estimation of the score from the node n to a leaf, here $f(n) \equiv 0$, the constant zero function. Note that h is a monotone function.

Remark 2. Let $Q_T = \infty$. If Q is implemented as a stack, then GTT traverses the tree in a Depth-first manner. If implemented as a First-In-First-Out queue, GTT will function as a Breadth-first search.

Remark 3. Note that, if the query spectrum is compared to an incorrect peptide sequence, then it doesn't really matter whether or not we lose the best goal.

Fast Match heuristics. In the process of database search, the experimental spectrum q is compared to peptide p (from the peptide database), if $|PM(q) - PM(p)| < \Delta$, where Δ is a precursor mass tolerance (usually Δ <1 Dalton).

PTMs can add or subtract several tens or hundreds Daltons with respect to the precursor mass of the experimental spectrum, thus the precursor mass tolerance needs to be widened. In this way a query is compared to a very large of number peptides that will substantially increase the execution time.

PTMSearch includes an additional filtering step we term Fast Match, wherein a theoretical peptide t is compared to the experimental spectrum q if $SPC(q, t) > F$, where F is a given threshold. If we set $F = 0$, this technique can miss the correct peptide if both the first and the last amino acids of the peptide carry PTMs because in this case all the fragment ions are shifted.

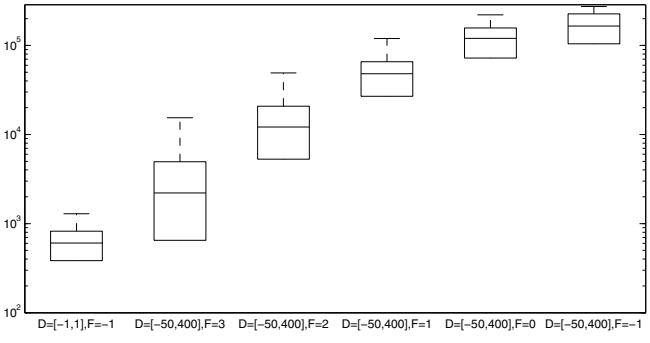


Fig. 4. Boxplot diagram about the number of the peptides to be processed per each spectrum

Figure 4 shows the number of the peptides to be compared with the query, at various parameter settings. Simply put, widening of the mass range from D=[-1,1] to D= [-50,400] increases the number of the peptide comparisons by two orders of magnitude. However, the application of Fast Match can significantly reduce the number of comparisons. The data in the plot were calculated on the experimental dataset described in chapter 6.

5 Significance Calculation of a Hit

In the literature, the number of the random matches between two spectra are modeled either by Poisson or by hypergeometrical distributions [10, 11]. The search for PTMs can be pictured as a generating all modified theoretical spectra, comparing them to the experimental spectrum, and picking the one with the maximum score. It is known that the distribution of the maximum of random

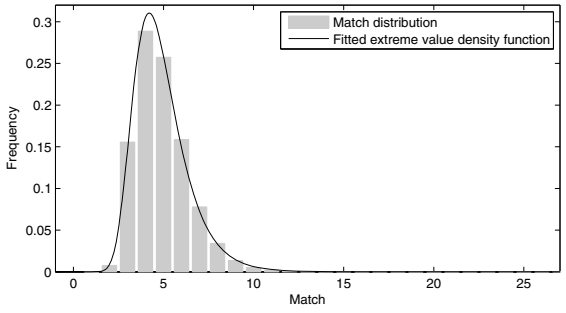


Fig. 5. Distribution of the matches on a real dataset. The solid line is a fitted extreme value probability density function on the experimental data. The number of the allowed PTMs is set to 3.

numbers tends to an extreme value distribution [12]. Thus, we use an extreme value distribution to calculate the statistical significance (p-value) of the hit to measure the probability the hit occurred by chance.

In Figure 5 we compared a real (experimental) spectrum against the decoy dataset, where the limit of the PTMs $K = 3$. The figure shows that the extreme value distribution fits the data well.

6 Experimental Results

For the experimental tests we used MatLab (version R2010b)as a frame to load the data and evaluate the results. PTMSearch (and GTT) was implemented in C++ and used as a module in MATLAB. Computational experiments were carried out on a Linux cluster with 20 nodes and one frontend node, each with 2.2Ghz CPU and 1GByte memory.

IPIHuman (version 3.71) downloaded from `http://www.ebi.ac.uk/IPI/IPIhelp.html` was used as the protein sequence database (86309 sequences). These sequences were cut into peptides using the tryptic digestion rule in the Cleavage routine of Matlab's Bioinformatic Toolbox. This *in silico* digestion resulted in 727707 unique peptides. For the calculation or theoretical fragment ions we used the program THEOSPEC 2.0 (available at: `http://sourceforge.net/projects/protms/files/theospec/`)

The list of modifications were downloaded from the OMSSA project site (`http://www.ncbi.nlm.nih.gov/IEB/ToolBox/CPP_DOC/lxr/source/src/algo/ms/omssa/mods.xml`) on Dec. of 2010. This file contains 207 records of PTMs. We have considered only chemical modifications of amino acids. Therefore the current iteration does not take into account N- and C-terminal modifications, such as might be found by the removal of the initiator methionine. Thus, we used PTM records that are tagged by "MSModType Value = 0" in the mods.xml file. In the end we obtained 142 PTMs. The distribution of the number of the modifications per amino acids can be found in Figure 3.

The precursor mass tolerance was set to $\Delta = 0.4$, the fragment ion match tolerance was $\delta = 0.4$, and PTMs mass range was set to $D = [-50, 400]$ Da. The Fast Match parameter was $F = 1$. The size of the queue Q was $Q_T = 50 * \log(n)$ based on the author's observations.

For statistical significance calculation we used a decoy dataset consisting theoretical spectra calculated from reversed peptide sequences [13, 14]. During database search, the query was first compared with the theoretical peptide dataset, and then with the decoy dataset [13]. For the evaluation, we plotted the number of the positives as a function of the False Positive Rate (FPR) at various thresholds.

6.1 Results on a Toy Datasets

We generated *toy dataset* by randomly selecting 200 peptides from the theoretical database and generating the corresponding theoretical fragment ion peaks. Then, we randomly added 2-3 PTMs from our list to each of the peptide spectra. This dataset was then searched against the peptide database with PTMSearch. We found that our method found all the generated modifications (Data not shown).

For an illustration we examined how the limit K of the PTMs interferes with the expected number of the random matches. We found that, as expected, increasing the limit of PTMs increases the number of the random matches, which in turn lessens the statistical significance of the true hit. This is shown in Figure 6. However, this effect could be avoided by increasing the instruments accuracy, thereby decreasing the ion match tolerance δ parameter.

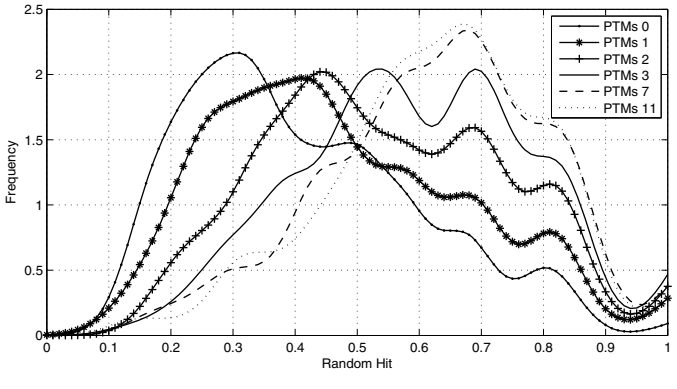


Fig. 6. The distribution of the random hits as a function of the PTMs limit K on the toy dataset. Here the number of the matches were divided the length of the peptide as a normalization.

6.2 Calculations on Real Data

The Aurum dataset has been designed to provide a standard, manually curated dataset of experimental spectra for comparison and evaluation of newly

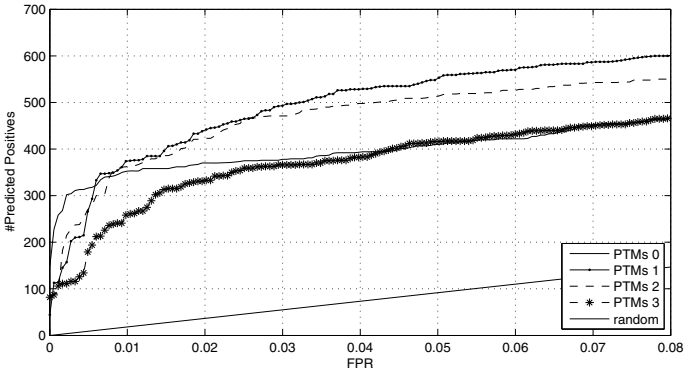


Fig. 7. The number of identified peptides at various PTM limit as a function of FPR

developed algorithmic approaches and is freely available [15]. The dataset contains 1834 MS/MS spectra obtained on 246 known, individually purified and trypsin-digested protein samples recorded on MALDI TOF/TOF instruments. The spectra were preprocessed as described in OMSSA [11].

The Figure 7 show a plot of the number of the positively predicted spectrum as a function of the FPR on the Aurum dataset. Table 1 shows detailed results obtained with PTMSearch at FPR=1% and FPR=5% levels. Columns show the number of the identified peptides at various PTM limits, while rows show the distribution of singly-, doubly-, and triply-modified peptides, respectively. For example, PTMSearch has identified 514 peptides at 5% FPR level, of which 243 peptides contained no modifications, 180 peptides were tagged by exactly 1 PTM, and 91 peptides were tagged by exactly 2 PTMs.

Table 1. The number of the peptides identified with PTMSearch at 1% and 5% of FPR respectively

FPR	PTMs	0	1	2	3
1%	sum	352	374	362	258
	0	352	240	203	138
	1		134	113	62
	2			46	28
	3				30
5%	sum	410	553	514	416
	0	410	295	243	194
	1		258	180	105
	2			91	42
	3				75
Time(min)		11	120	754	1281

1281 minutes is approx. 21 hours.

Table 2. Results obtained with MS-Alignment. PTMs limit is 1.

FPR	PTMs 0	PTMs 1	sum	Time
5%	196	91	287	5.5 days

PTMSearch reported an unexpectedly large number of propionamide modifications of cysteine in the Aurum dataset. Including triply modified peptides PTMSearch found 138 cysteines that carry this Acrylamide adduct. We then used X!Tandem [16] to perform targeted search for this modification and could only validate 58 propionamide modifications. This data further supports the importance of untargeted search for PTM discovery.

This filtered Aurum dataset was submitted to the online version of MS-Alignment at `http://proteomics.ucsd.edu/LiveSearch/` [7]. Here the precursor mass tolerance and the ion tolerance was the same as in the PTMSearch case. The database was the IPI.Human (v.3.81), The modification mass range was set to $D = [-200, 200]$. Results are presented in the Table 2. Note that, the run time cannot be compared to PTMSearch run time since neither the system architecture nor the load of the MS-Alignment server were known. Limit of PTMs was set to 1. MS-Alignment has identified 287 peptides with at most one PTM, while PTMSearch identified 553 peptides under same conditions.

7 Discussion, Future Plans

Protein identification based on tandem mass spectrometry and database search is fast growing field in continuous need of new computational approaches. The untargeted PTM search algorithm described targets one of the most difficult problems in this field, identification of post-translationally modified proteins.

The novel features of the approach are the tree-representation of the search space, the greedy heuristics based on chemical rules and the speedup techniques that allow one to bring down the computational speed to a manageable level. We investigated the computational properties of the heuristics and showed that the limits imposed on PTMs decrease the search space to polynomial. We gave an estimate to the error introduced by the greedy heuristics and presented a significance calculation scheme appropriate for this algorithm.

The current work was undertaken in order to demonstrate the feasibility of the tree-traversal approach. There are many further improvements possible, for instance one can use spectrum similarity measures instead of the simple SPC employed here. The current algorithm is apparently not substantially faster as compared to other PTM identification methods, however an exact comparison could not be made because of code availability problems. Nevertheless, a substantial speedup can be expected upon massive parallelization or porting the algorithm to GPU processors, since tree-traversal algorithms are known from this perspective. We plan to include the algorithm into a free online search engine where those computational aspects that were outside the scope of this work (data filtering, protein inference etc.) will also be dealt with.

References

1. Yates, J.R., Eng, J.K., McCormack, A.L., Schieltz, D.: Method to correlate tandem mass spectra of modified peptides to amino acid sequences in the protein database. Analytical Chemistry 67(8), 1426–1436 (1995)
2. Nathalie, F.-M., Garavelli, J.S., Boeckmann, B., Duvaud, S., Gasteiger, E., Gateau, A., Veuthey, A.-L., Bairoch, A.: Annotation of post-translational modifications in the Swiss-Prot knowledge base. PROTEOMICS 4(6), 1537–1550 (2004)
3. Yan, B., Zhou, T., Wang, P., Liu, Z., Emanuele II, V.A., Olman, V., Xu, Y.: A Point-Process Model for Rapid Identification of Post-Translational Modifications. Pacific Symposium on Biocomputing (11), 327–338 (2006)
4. Nesvizhskii, A.I., Vitek, O., Aebersold, R.: Analysis and validation of proteomic data generated by tandem mass spectrometry. Nature Methods 4(10), 787–797 (2007)
5. Li, Y., Chi, H., Wang, L.-H.H., Wang, H.-P.P., Fu, Y., Yuan, Z.-F.F., Li, S.-J.J., Liu, Y.-S.S., Sun, R.-X.X., Zeng, R., He, S.-M.M.: Speeding up tandem mass spectrometry based database searching by peptide and spectrum indexing. Rapid communications in mass spectrometry: RCM 24(6), 807–814 (2010)
6. Ahrné, E., Müller, M., Lisacek, F.: Unrestricted identification of modified proteins using MS/MS. Proteomics 10(4), 671–686 (2010)
7. Tsur, D., Tanner, S., Zandi, E., Bafna, V., Pevzner, P.A.: Identification of post-translational modifications via blind search of mass-spectra. Nat. Biotechnol. 23, 1562–1567 (2005)
8. Baliban, R.C., DiMaggio, P.A., Plazas-Mayorca, M.D., Young, N.L., Garcia, B.A., Floudas, C.A.: A Novel Approach for Untargeted Post-translational Modification Identification Using Integer Linear Optimization and Tandem Mass Spectrometry. Molecular & Cellular Proteomics 9(5), 764–779 (2010)
9. Knuth, D.E.: The art of computer programming, 2nd edn., vol. 3. Addison-Wesley Longman Publishing Co., Amsterdam (1998)
10. Sadygov, R.G., Yates, J.R.: A hypergeometric probability model for protein identification and validation using tandem mass spectral data and protein sequence databases. Anal. Chem. 75(15), 3792–3798 (2003)
11. Geer, L.Y., Markey, S.P., Kowalak, J.A., Wagner, L., Xu, M., Maynard, D.M., Yang, X., Shi, W., Bryant, S.H.: Open Mass Spectrometry Search Algorithm (June 2004)
12. Anderson, C.W.: Extreme value theory for a class of discrete distributions with applications to some stochastic processes. Journal of Applied Probability 7, 99–113 (1970)
13. Käll, L., Storey, J.D., MacCoss, M.J., Noble, W.S.S.: Assigning significance to peptides identified by tandem mass spectrometry using decoy databases. Journal of proteome research 7(1), 29–34 (2008)
14. Fenyo, D., Beavis, R.C.: A method for assessing the statistical significance of mass spectrometry-based protein identifications using general scoring schemes. Analytical Chemistry 75(4), 768–774 (2003)
15. Falkner, J.A., Kachman, M., Veine, D.M., Walker, A., Strahler, J.R., Andrews, P.C.: Validated maldi-tof/tof mass spectra for protein standards. Journal of the American Society for Mass Spectrometry 18(5), 850–855 (2007)
16. Craig, R., Beavis, R.C.: Tandem: matching proteins with tandem mass spectra. Bioinformatics 20(9), 1466–1467 (2004)

Efficient Mining of Top Correlated Patterns Based on Null-Invariant Measures

Sangkyum Kim[1], Marina Barsky[2], and Jiawei Han[1]

[1] University of Illinois at Urbana-Champaign, Urbana, IL, USA
{kim71,hanj}@illinois.edu
[2] Simon Fraser University, BC, Canada
marina_barsky@sfu.ca

Abstract. Mining strong correlations from transactional databases often leads to more meaningful results than mining association rules. In such mining, null (transaction)-invariance is an important property of the correlation measures. Unfortunately, some useful null-invariant measures such as $Kulczynski$ and $Cosine$, which can discover correlations even for the very unbalanced cases, lack the (anti)-monotonicity property. Thus, they could only be applied to frequent itemsets as the post-evaluation step. For large datasets and for low supports, this approach is computationally prohibitive. This paper presents new properties for all known null-invariant measures. Based on these properties, we develop efficient pruning techniques and design the Apriori-like algorithm NICoMiner for mining strongly correlated patterns *directly*. We develop both the threshold-bounded and the top-k variations of the algorithm, where top-k is used when the optimal correlation threshold is not known in advance and to give user control over the output size. We test NICoMiner on real-life datasets from different application domains, using $Cosine$ as an example of the null-invariant correlation measure. We show that NICoMiner outperforms support-based approach more than an order of magnitude, and that it is very useful for discovering top correlations in itemsets with low support.

1 Introduction

One of the central tasks in data mining is finding correlations in binary relations. Typically, this is formulated as a *market basket* problem [2], where there is a set of baskets (*transactions*), each of which is a set of items purchased together. The goal is to find correlations between items, based on their recurrent co-appearance in the same transaction. The usefulness of the correlations based on the market-basket concept was demonstrated in many different application domains such as climate studies [18], public health [5], or bioinformatics [9,21]. With the trend of collecting more and more digitized data, the discovery of meaningful correlations offers a new insight into relationships between objects in these large data collections.

In this paper we study the problem of finding groups of items with the top correlations for a given dataset. This implies that we need to rank the correlations. There is no canonical way to assess the degree of the correlation. This seems to be problem-specific and cannot be captured by a single correlation measure which is the best for all cases. As a result, a number of correlation measures has been proposed [8,16,17,19].

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 177–192, 2011.

Table 1. The same dataset contains coffee c, milk m, popcorn p, and soda s. The total number of transactions is $N = 100,000$. According to $Lift$, correlation(p, s) is significantly stronger than correlation(m, c). Assessed by null-invariant measures, correlation(m, c) is always much stronger than correlation(p, s), which is more meaningful, since cm occur together in much more transactions than ps.

mc	$\bar{m}c$	$m\bar{c}$	$\bar{m}\bar{c}$	$Lift(m, c)$	$Cosine(m, c)$
10,000	1,000	1,000	88,000	8.26	0.91
ps	$\bar{p}s$	$p\bar{s}$	$\bar{p}\bar{s}$	$Lift(p, s)$	$Cosine(p, s)$
1,000	1,000	1,000	97,000	25.00	0.50

In this work we limit ourselves to *null (transaction)-invariant* [8,16,17,19] correlation measures based on conditional probabilities. They quantify the degree of mutual relationships between items in a group without taking into account the items outside the group in question. For example, if we are computing the correlation between coffee (c) and milk (m), a null-invariant measure does not depend on the number of transactions which contain neither coffee nor milk - *null transactions* with respect to c and m. Thus, these measures are *null (transactions)-invariant*.

The importance of null-invariance for uncovering meaningful relationships between objects was analyzed in [19]. If we use correlation measures which are not null-invariant, the relationships between objects may appear or disappear simply by changing the number of transactions which do not contain items in question.

Even for ranking correlations within *the same* dataset we cannot rely on expectation-based (not null-invariant) measures, since they produce inconsistent and controversial results, as shown in a sample dataset, presented in Table 1. Here the degree of the correlation of two pairs of items is assessed by $Lift$ (not null-invariant) and by $Cosine$ (null-invariant). The items in pair (c, m) are intuitively more correlated than in (p, s), since they occur together in 83% of all transactions with c or m, while (p, s) occur together only in 33%. This is reflected in $Cosine$ values 0.91 and 0.50 respectively. However, according to $Lift$, correlation in pair (p, s) is significantly larger than in (c, m), which contradicts our intuition and the common sence. Hence, in order to produce meaningful and consistent top correlations we require from the correlation measure to be null-invariant.

The five known null-invariant correlation measures are *All Confidence*, *Coherence*, *Cosine*, *Kulczynski* and *Max Confidence* [19]. The degree of the correlation is represented as a real number between 0 and 1.

For different datasets, the strongest correlations may have different values. It is not always appropriate to set a correlation threshold such as 0.5 for all datasets. Hence, it is important to be able to mine the top correlated patterns, instead of patterns with correlation larger than a given threshold. This leads to a problem of mining top-k null-invariant correlations. An example of top-10 correlations, which we extracted from the titles of the database-related publications [15], is shown in Table 2. Note that the correlation here is not expected to be very high, since people use different word combinations to describe even similar ideas. Nevertheless, the top correlated patterns represent quite meaningful concepts.

Table 2. Top-10 highly correlated term groups from the paper titles in the DB-DM-IR subset [15] of the DBLP dataset [1] with minimum support $\theta = 0.02\%$

	Pattern	Support	$Cosine$
1	$object, orient, database$	748	0.19
2	$sense, word, disambiguation$	26	0.18
3	$support, vector, machine$	122	0.17
4	$enforcement, law, coplink$	7	0.16
5	$nearest, neighbor, search$	74	0.13
6	$reverse, nearest, neighbor$	23	0.13
7	$server, sql, microsoft$	25	0.12
8	$retrieval, cross, language$	187	0.11
9	$model, relationship, entity$	139	0.11
10	$random, field, conditional$	13	0.10

Finding the itemsets with the highest correlations is not trivial. The naïve approach would be to extract all frequent itemsets, and then to rank them based on the correlation within each frequent itemset. Unfortunately, this approach is valid only for itemsets with high support, and in this case the discovered correlations mostly represent the common knowledge. If we are to discover interesting correlations in itemsets with low support, the number of such itemsets can reach several thousands or even millions, thus making the post-evaluation approach computationally infeasible. In addition, the degree of the correlation between items can be higher in itemsets with lower support. This is especially true for such problems as finding correlations between words or finding correlations between authors in a publication database. Therefore, we want to design an efficient framework in which we would be able to find the groups of the top correlated items with low support, without first collecting all frequent itemsets.

The algorithms for the direct mining of interesting null-invariant patterns exist. For example, the direct computation based on *All Confidence* and *Coherence* was proposed in [10]. However, it is applicable only for null-invariant measures which have the *anti-monotonicity* property. Out of five measures, only *All Confidence* and *Coherence* are anti-monotonic. Unfortunately, using only *All Confidence* or *Coherence* may not be appropriate for cases involving unbalanced supports, which was demonstrated in [19]. Strong correlations for such unbalanced cases can be captured if we evaluate the relationships as an *average* of conditional probabilities. For such cases, two measures $Cosine$ and $Kulczynski$ are the most appropriate ones.

Both $Cosine$ and $Kulczynski$ represent the means of conditional probabilities: the geometric mean and the arithmetic mean, respectively. For an itemset $A = \{a_1, \cdots, a_n\}$:

$$Cosine(A) = \sqrt[n]{\prod_{i=1}^{n} P(A|a_i)}, \text{ and } Kulczynski(A) = \frac{1}{n}\sum_{i=1}^{n} P(A|a_i)$$

where $P(A|a_i)$ is a conditional probability of A given a_i.

Being an average, $Cosine$ and $Kulczynski$ do not possess neither monotonicity nor anti-monotonicity properties, and the Apriori principle cannot be applied for efficient pruning based on these measures. Hence, the discovery of all patterns with high $Cosine$

and $Kulczynski$ values poses a great computational challenge, especially for itemsets with low support. To solve this challenging problem, we develop an efficient algorithmic framework based on new pruning properties *common to all null-invariant measures*, but especially valuable for $Cosine$ and $Kulczynski$.

Specifically, our study makes the following contributions.

1. We discover new mathematical properties common to all null-invariant measures.
2. Based on these properties, we design a new pruning strategy which relies mainly on correlation measures rather than on support.
3. We propose new algorithm NICoMiner for *Null Invariant Correlation Mining* and demonstrate its high efficiency on a wide variety of synthetic and real-life datasets.
4. In order to make NICoMiner self-adjustable to the level of the correlations existing in different datasets, and to give user the control over an output size, we develop the top-k version of NICoMiner, which allows us to find the top-k correlated itemsets without specifying the correlation threshold.
5. Finally, we show meaningful correlations discovered by NICoMiner in itemsets with low support. It is hard or somtetimes impossible to find such correlations using the support-based method alone.

The remainder of the paper is organized as follows. In Section 2 we formally define correlated patterns. In Section 3 we describe the new properties of null-invariant measures, and in Section 4 we present our new algorithm. Section 5 is a detailed report on our experiments with synthetic and real datasets. Related work is presented in Section 6, followed by conclusions and future work in Section 7.

We start by introducing a few concepts. Note that for the rest of the paper we use $Cosine$ as a representative of null-invariant correlation measures.

2 Preliminaries

Let $\mathcal{I}$ be a set of items. We define an *itemset* $A = \{a_1, \ldots, a_n\}$ to be a subset of n items from $\mathcal{I}$. Let $\mathcal{T}$ be a set of transactions where each *transaction* is a subset of $\mathcal{I}$. The *support* of an itemset A, $sup(A)$, is defined to be the number of transactions containing all items in A. An itemset A is *frequent* if its support $sup(A)$ is no less than a user-defined *minimum support threshold* θ.

Cosine in terms of supports is explicitly defined as:

$$cos(A) = \frac{sup(A)}{\sqrt[n]{sup(a_1) \times \cdots \times sup(a_n)}}. \tag{1}$$

We define the correlation between items in an itemset as follows:

Definition 1. *An itemset $A = \{a_1, \ldots, a_n\}$ is correlated if $cos(A) \geq \gamma$ for a given minimum correlation threshold γ.*

The *problem of threshold-based correlation mining* is to find all correlated itemsets for the correlation threshold γ. But, even for the experts, it is sometimes hard to set the proper value of γ. For such cases, it would be helpful to know several patterns with the highest correlation values. This is the *problem of top-k correlation mining*, where

Table 3. A small transactional database of 6 transactions and 6 items

TID	Transaction
T_1	a_1, a_3, a_4, a_5, a_6
T_2	a_3, a_5, a_6
T_3	a_2, a_4
T_4	a_1, a_4, a_5, a_6
T_5	a_3, a_6
T_6	a_2, a_4, a_5

only k patterns with the highest correlation values are presented to the user. Note that a minimum correlation threshold γ is not required for top-k correlation mining.

The lack of the anti-monotonicity property for $Cosine$ poses significant challenges for mining top correlated patterns. This can be illustrated by the following example.

Example 1. Consider small database of transactions shown in Table 3.
Correlation value for 2-itemset $X = \{a_4, a_6\}$ is $cos(X) = 0.50$. 3-itemset $X' = \{a_1, a_4, a_6\}$ is a superset of X, and its correlation is $cos(X') = 0.63$. Thus, $Cosine$ is *not anti-monotonic*. For the correlation threshold $\gamma = 0.60$, we cannot prune all supersets of X, even though the correlation in X is below γ.

Correlation value for 2-itemset $Y = \{a_1, a_4\}$ is $cos(Y) = 0.71$. 3-itemset $Y' = \{a_1, a_4, a_5\}$ is a superset of Y, and its correlation is $cos(Y') = 0.63$. Thus, $Cosine$ is also *not monotonic*. Knowing that Y is a correlated itemset, we cannot assume that all supersets of Y are also correlated. This shows that finding that $cos(X) < \gamma$ or that $cos(Y) \geq \gamma$ does not tell us anything about the correlation value in their supersets, and hence we cannot stop the extension of X or Y to larger itemsets. ■

3 New Properties of Null-Invariant Measures

In this section, we describe useful mathematical properties, common to all known null-invariant measures. These properties are the basis for an efficient pruning used in the NICoMiner algorithm. Our framework is based on the level-wise Apriori algorithm, where each level n corresponds to itemsets of n items.

3.1 Level-Based Properties

The relationships between $Cosine$ of n-itemset A and $Cosine$ values of all its subsets of size n-1 are captured by the following lemma:

Lemma 1. *For any n-itemset* $A = \{a_1, \cdots, a_n\}$ *and a set* $\mathcal{S}$ *of all A's (n-1)-subitemsets:*

$$cos(A) \leq \max_{B \in S}(cos(B)). \tag{2}$$

Proof. Since the maximum is not smaller than the geometric mean:

$$\max_{B \in S}(\cos(B)) \geq \sqrt[n]{\cos(a_1, \cdots, a_{n-1}) \times \cdots \times \cos(a_2, \cdots, a_n)}. \tag{3}$$

Then by the definition of $Cosine$ and from the anti-monotonicity of support:

$$\max_{B \in S}(\cos(B)) \tag{4}$$

$$\geq \sqrt[n]{\frac{\sup(a_1, \cdots, a_{n-1})}{\sqrt[n-1]{\sup(a_1) \times \cdots \times \sup(a_{n-1})}} \times \cdots \times \frac{\sup(a_2, \cdots, a_n)}{\sqrt[n-1]{\sup(a_2) \times \cdots \times \sup(a_n)}}} \tag{5}$$

$$\geq \frac{sup(a_1, \cdots, a_n)}{\sqrt[n]{sup(a_1) \times \cdots \times sup(a_n)}} \tag{6}$$

$$= \cos(A). \tag{7}$$

■

Lemma 1 presents an upper bound of *Cosine* in terms of *Cosine* values of subitemsets. A simple corollary follows from Lemma 1: once $Cosine$ values of all (n-1)-subitemsets of $A = \{a_1, \cdots, a_n\}$ are less than γ, $cos(A) < \gamma$. However, this does not mean that A and its supersets can be pruned. There might be a superset of A, $A' = \{a_1, \cdots, a_n, a_{n+1}\}$ with $cos(A') \geq \gamma$, because the condition of the lemma may not be satisfied due to the newly added item a_{n+1}.

Nevertheless, Lemma 1 leads to a simple condition for the termination of correlation pattern growth. Even though *Cosine* for individual patterns is not anti-monotonic, there is a level-based property which we for convenience call *level-anti-monotonicity*. Namely, if all patterns at level n have $Cosine$ values less than γ, then all their supersets have $Cosine$ less than γ.

Let $\mathcal{I}_n$ be set of all n-itemsets at level n. We denote the maximum cosine value for all itemsets in $\mathcal{I}_n$ by $maxCos(\mathcal{I}_n)$. We prove that:

Theorem 1. *Cosine* is *level-anti-monotonic.*

Proof. Let $\mathcal{I}_{n+1}$ be set of all (n+1)-itemsets at level n+1, and let A' be an itemset from $\mathcal{I}_{n+1}$ with maximum cosine value. Let A be an n-subitemset of A' whose cosine value is the maximum from all n-subitemsets of A'. Then, by Lemma 1,

$$maxCos(\mathcal{I}_n) \geq cos(A) \geq cos(A') = maxCos(\mathcal{I}_{n+1}). \tag{8}$$

■

From Theorem 1 follows:

Corollary 1. Termination of pattern growth (TPG)
If all itemsets at level n are not correlated, then all itemsets at level n' are not correlated for any $n' \geq n$.

Note that TPG holds for all five null-invariant correlation measures. The proofs are essentially similar to that of Cosine, and we omit them due to the page limit.
To demonstrate the termination of pattern growth, consider the following example.

Example 2. For a database described in Table 3 with the *minimum support threshold* $\theta = 2$, there exist 5 frequent 3-itemsets shown in Table 4. Assuming the *minimum correlation threshold* $\gamma = 0.75$, all 3-itemsets have correlation below the threshold. Then, based on TPG, we do not need to mine n-itemsets for $n \geq 3$, and therefore pattern growth terminates. ■

Table 4. *Cosine* values for all five frequent 3-itemsets from the database in Table 3 ($\theta = 2$). If $\gamma = 0.75$, we can terminate correlation pattern growth according to TPG.

Pattern	a_1, a_4, a_5	a_1, a_4, a_6	a_1, a_5, a_6	a_3, a_5, a_6	a_4, a_5, a_6
Cosine	0.63	0.63	0.63	0.55	0.5

3.2 Properties Based on a Single Item

Since *Cosine* is not anti-monotonic, we cannot prune n-itemset A even if A is not correlated. But, in the following, we claim that for some item a from $\mathcal{I}$, knowing correlation values of all (n-1)-itemsets containing a allows to prune n-itemsets containing a.

Lemma 2. *For n-itemset $A = \{a_1, \cdots, a_n\}$, and all its subsets of size n-1 which share the same single item a, if (1) all these subsets are not correlated and (2) the support of at least one item $a_i \neq a$ in A is greater than or equal to $sup(a)$, then A cannot be correlated.*

Proof. Assume $a_1 = a$ and $sup(a_n) = \max\{sup(a_1), \cdots, sup(a_n)\}$, without loss of generality. By simple algebra, we can show that

$$\sqrt[n-1]{sup(a_1) \times \cdots \times sup(a_{n-1})} \leq \sqrt[n]{sup(a_1) \times \cdots \times sup(a_n)}. \tag{9}$$

Then

$$\cos(A) = \frac{sup(A)}{\sqrt[n]{sup(a_1) \times \cdots \times sup(a_{n-1}) \times sup(a_n)}} \tag{10}$$

$$\leq \frac{sup(A)}{\sqrt[n-1]{sup(a_1) \times \cdots \times sup(a_{n-1})}} \tag{11}$$

$$\leq \frac{sup(A - \{a_n\})}{\sqrt[n-1]{sup(a_1) \times \cdots \times sup(a_{n-1})}} \tag{12}$$

$$\leq \cos(A - \{a_n\}) \tag{13}$$

$$< \gamma, \tag{14}$$

where $A - \{a_n\}$ represents the (n-1)-subitemset of A which does not contain an item a_n with the maximum support. ∎

In other words, if we know that all sub-itemsets containing item a are not correlated, we know that adding another item cannot make any of them correlated, given this new item has support not less than $sup(a)$.
Based on Lemma 2, we can claim the following theorem:

Theorem 2. *Let item a be an item with the smallest support among all single items in the database. If all itemsets at level n containing a are not correlated, then all n'-itemsets containing a are not correlated for all $n' \geq n$.*

Proof. Each $(n+1)$-itemset A' which contains a can be thought of as an extension of some n-itemset containing a with an item a_{n+1}, which has the largest support among all the items in A' (since we know that support of a is not the largest). Then, by Lemma 2, $cos(A') < \gamma$. Since all n-itemsets containing item a have *Cosine* value less than γ, all

Table 5. Frequent 2-itemsets from the database in Table 3 ($\theta = 2$). For $\gamma = 0.75$, all supersets of a_1 and a_2 are not correlated according to SIBP.

Pattern	a_1, a_4	a_1, a_5	a_1, a_6	a_2, a_4	a_3, a_5	a_3, a_6	a_4, a_5	a_4, a_6	a_5, a_6
$Cosine$	0.71	0.71	0.71	0.71	0.58	0.87	0.75	0.5	0.75

$(n+1)$-itemsets containing item a have $Cosine$ value less than γ. Iteratively applying Lemma 2, now to extension of $(n+1)$-itemsets into $(n+2)$-itemsets, containing a, we conclude that none of the n'-itemsets containing a is correlated, for $n' \geq n$ ∎

Based on Theorem 2, we can derive a condition for pruning patterns which contain the same single item a. For convenience, we call the pruning of a non-promising single item and its supersets at level n the *single-item-based pruning* (SIBP).

Corollary 2. Single-Item Based Pruning (SIBP)
If the maximum $Cosine$ value for n-itemsets containing item a is less than γ, and a has the smallest support between single items existing in the database, then all n'-itemsets containing a can be pruned for $n' \geq n$.

For the level-wise processing, which we use here, such an item can be removed from the database. After removing it, we have a new, smaller database, and we can apply the same principle to the next item, which has the smallest support in this new database.

Again, SIBP holds for all null-invariant correlation measures. We skip the proofs due to the page limit, but the proofs are very similar or easier than that for $Cosine$.

The application of the SIBP principle can be illustrated on the following example.

Example 3. Consider the sample database in Table 3 ($\theta = 2$, $\gamma = 0.75$). First, all single frequent items $a_1 \ldots a_6$ are sorted by support. Then, while counting itemsets at level 2, the maximum $Cosine$ value of 2-item supersets of each a_i is recorded. For this example, we have: a_1 (sup:2, maxCos:0.71), a_2 (sup:2, maxCos:0.71), a_3 (sup:3, maxCos:0.87), a_4 (sup:4, maxCos:0.75), a_5 (sup:4, maxCos:0.75), and a_6 (sup:4, maxCos:0.86). Now, based on the SIBP principle, we can safely prune all 2-itemsets containing item a_1 (or item a_2), and we do not need to generate the following 3-itemsets in Table 4: $\{a_1, a_4, a_5\}$, $\{a_1, a_4, a_6\}$, and $\{a_1, a_5, a_6\}$. ∎

4 NICoMiner Algorithm

The general framework of NICoMiner is an Apriori-like level-wise (breadth-first) computation. The candidate itemsets for level n are generated from the itemsets on level n-1. Then the support and $Cosine$ are computed for all candidate n-itemsets, and they are pruned based on support and SIBP. The remaining n-itemsets are the candidates for the next level $n+1$. If all patterns at level n are not correlated, the algorithm terminates (TPG).

4.1 Threshold-Based Correlation Mining

Here we present the correlation mining algorithm (Algorithm 1) for the case when a minimum correlation threshold γ is given. The pruning properties developed in the

Algorithm 1. The threshold-based version of the NICoMiner Algorithm

input : a transactional database $\mathcal{D} = \{T_1, T_2, ..., T_n\}$, minimum correlation threshold γ, minimum support threshold θ
output: all patterns with correlation at least γ

```
1 scan D and find all frequent 1-itemsets I_1;
2 for n = 2, ··· do
3     generate candidate itemsets I_n from I_{n-1};
4     scan D to compute support and Cosine values of itemsets in I_n;
5     output frequent n-itemsets with Cosine ≥ γ;
6     prune itemsets from I_n based on SIBP and support;
7     if (maxCos(I_n) < γ) OR (no frequent n-itemsets) then break;
8 end
```

previous section allow to prune uncorrelated patterns in addition to the non-frequent patterns. In practice, the pruning power of TPG and SIBP is extremely high, which allows setting very low support thresholds.

4.2 Top-k Correlation Mining

Without knowing what is the top level of correlations in a given dataset, it is hard to choose an appropriate correlation threshold γ. Running the top-k version of our algorithm helps in this situation. After this, the information about the top correlations can be used to set a meaningful threshold in order to collect all interesting patterns. Often, the set of the top-k correlated patterns is interesting in its own right.

In order to find top-k correlated patterns, we can iteratively run the threshold-based NICoMiner until it produces at least k patterns. If in the current iteration the size of the output is less than k, we can decrease the correlation threshold γ and run Algorithm 1 with this new parameter. We implemented this iterative top-k approach, halving the correlation threshold in each iteration.

However, guessing the correlation threshold γ which produces close to k patterns is not efficient. Not only we need to repeat the entire computation several times, but if we accidentally set γ too low, we have an expensive computation and a huge output, while we are interested only in k patterns.

Much more efficient approach would be to adjust threshold γ throughout the mining process until we get top-k correlated patterns (Algorithm 2). Here, instead of using a fixed threshold value, we start with $\gamma = 0.0$ and keep top k correlated itemsets from the itemsets processed so far. Once we mine more than k patterns, we set γ to the k-th largest $Cosine$ value, and the pattern growth continues with this new, higher correlation threshold. Since the correlation threshold is constantly increasing, the termination of the pattern growth is reached earlier than in the method with the constant initial correlation threshold.

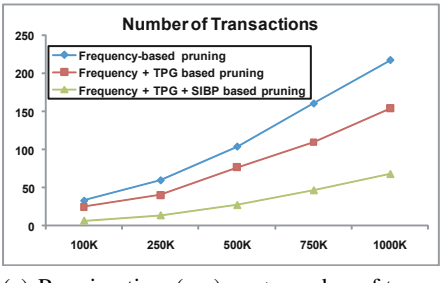
5 Experiments

In this section, we present experimental results for two versions of NICoMiner: one computes all patterns with the correlation above the minimum correlation threshold

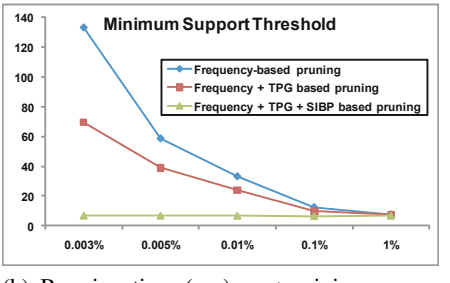
Algorithm 2. The top-k version of NICoMiner

input : a transactional database $\mathcal{D} = \{T_1, T_2, ..., T_n\}$, number k, minimum support threshold θ
output: set TOP of top-k correlated patterns

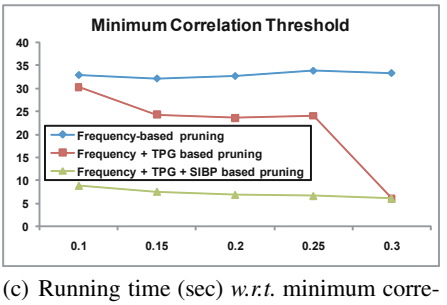
```
1  γ ← 0; TOP ← ∅;
2  scan D and find all frequent 1-itemsets I_1;
3  for n = 2, ··· do
4      generate candidate itemsets I_n from I_{n−1};
5      scan D to compute support and Cosine values of all candidate k-itemsets;
6      TOP ← TOP ∪ {correlated n-itemsets};
7      if |TOP| ≥ k then
8          keep only top-k in TOP;
9          γ ← minimum Cosine value in TOP;
10     end
11     prune itemsets from I_n based on SIBP and support;
12     if (maxCos(I_n) < γ) OR (no frequent n-itemsets) then break;
13 end
```



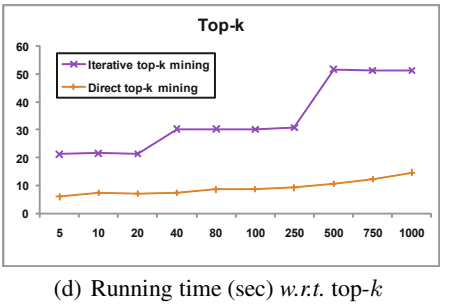
(a) Running time (sec) *w.r.t.* number of transactions



(b) Running time (sec) *w.r.t.* minimum support threshold



(c) Running time (sec) *w.r.t.* minimum correlation threshold

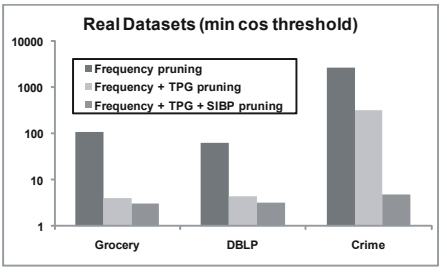


(d) Running time (sec) *w.r.t.* top-k

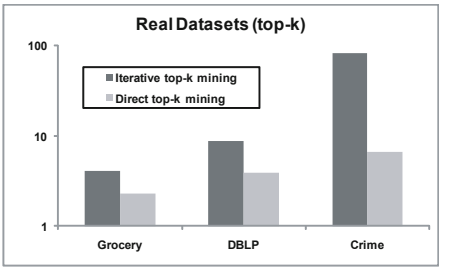
Fig. 1. Performance results for synthetic datasets

and the other finds the top-k correlations. All experiments were performed on a Linux (ver 2.6.18) server with quad core Xeon 5500 processors and 48 GB of main memory.

For the threshold-based version, we used the support-based pruning as the baseline. To evaluate the pruning power of each new technique, we added to the baseline



(a) Running time (sec) for threshold-based version.



(b) Running time (sec) for top-k version (k=100).

Fig. 2. Performance results for real datasets

algorithm the pattern growth termination (TPG), and then enhanced it with the single-item-based pruning (SIBP). The latter represents the full version of the threshold-based NICoMiner.

For the top-k version, we compared our direct top-k NICoMiner with the naïve iterative top-k mining, which uses multiple iterations of the threshold-based version, halving the correlation threshold in each iteration, until the output contains at least k patterns.

5.1 Synthetic Datasets

Synthetic datasets for our experiments were generated by the generator used in [14]. The default parameters are: number of transactions $N = 100K$, average number of items per transactions $W = 5$, number of distinct items $|\mathcal{I}| = 1K$. The default set of thresholds for all experiments is as follows: minimum support threshold $\theta = 0.01\%$, and minimum correlation threshold $\gamma = 0.2$.

For the correlation-based version of NICoMiner we show the dependence of the running time on the following parameters: number of transactions, minimum support threshold, and minimum correlation threshold.

Number of transactions: The results in Figure 1(a) show the comparative performance for 5 different synthetic datasets with number of transactions varying from 100K to 1M. For all methods, the running time shows linear dependency on N, which means that the size of a dataset is not the limiting parameter for the performance of NICoMiner.

Minimum support threshold: In Figure 1(b), we evaluated the performance of our algorithm for various minimum support threshold values. As the threshold becomes lower, frequency-based pruning deteriorates exponentially. Adding TPG makes the baseline algorithm about two times faster, but the performance still degrades for low support thresholds. On the other hand, the full version of NICoMiner demonstrates consistently high performance. For the lowest minimum support threshold 0.003%, our algorithm is more than an order of magnitude faster than two other methods. This demonstrates the main power of our algorithm, which is meant for finding correlated patterns with low supports.

Table 6. Examples of top correlated patterns for each dataset

Dataset	Pattern	*sup*	*cos*
GROCERIES	{*butter milk, yogurt*}	84	0.14
	{*salty snack, popcorn*}	22	0.14
	{*chocolate, candy*}	49	0.13
	{*frankfurter, brown bread*}	70	0.12
	{*sausage, white bread*}	71	0.12
DBLP AUTHORS	{*Steven M. Beitzel, Eric C. Jensen*}	25	1.00
	{*In-Su Kang, Seung-Hoon Na*}	20	0.98
	{*Ana Simonet, Michel Simonet*}	16	0.94
	{*Caetano Traina Jr., Agma J. M. Traina*}	35	0.92
	{*Claudio Carpineto, Giovanni Romano*}	15	0.91
COMMUNITIES	{*People with social security income:* $> 80\%$, *Age* ≥ 65*:* $> 80\%$}	47	0.76
	{*Large families* (≥ 6)*:* $\leq 20\%$*, White:* $> 80\%$}	1017	0.75
	{*In dense housing* (≥ 1 *per room*)*:* $> 80\%$, *Hispanic:* $> 80\%$*, Large families* (≥ 6)*:* $> 80\%$}	53	0.64
	{*People with Bachelor or higher degree:* $> 80\%$, *Median family income: very high* }	60	0.63
	{*People with investment income:* $> 80\%$, *Median family income: very high* }	66	0.61

Minimum correlation threshold: In Figure 1(c), we show the effect of the minimum correlation threshold. Frequency-based pruning does not depend on the minimum correlation threshold, since there is no pruning based on correlation values. The termination of pattern growth (TPG) cannot be applied before all correlations at some level has been evaluated. For the largest correlation threshold $\gamma = 0.3$, the algorithm terminates after level 2 (all 2-itemsets are below threshold), while for the lowest correlation threshold $\gamma = 0.1$, it continues up to level 4. This explains the difference in the running time. For $\gamma = 0.1$, the full NICoMiner also stops at level 4, however it generates much less candidates due to the high pruning power of SIBP.

Top-k: In Figure 1(d), we compare the iterative and the direct top-k correlation mining for various values of k. Both approaches used all pruning properties for maximum performance. As expected, the direct approach was faster than the iterative approach. The gap in performance becomes bigger as k grows. This is because more iterations are performed by the iterative method before the output contains at least k patterns.

5.2 Real Datasets

We tested NICoMiner applying the market basket concept to three real-life datasets. The performance results are presented in Figure 2. In Figure 2(a) we compare the efficiency of different pruning methods with the baseline pruning by support, and in Figure 2(b) we compare the direct top-k version with the iterative top-k mining.

1. The GROCERIES dataset [6,7] ($9,800$ transactions) represents 1-month of the point-of-sale transactions in the local grocery store. This dataset is comparatively

sparse: the number of frequent itemsets is low even for the minimum support threshold as low as 0.05%. Nevertheless, for $\theta = 0.05\%$ and $\gamma = 0.10$ our algorithm is 35 times faster than the baseline support-based computation. This performance gain for such relatively small dataset shows the potential of our method for typical market basket applications.

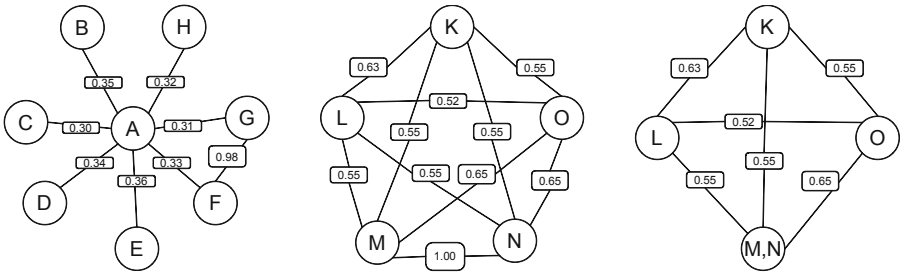
2. The DBLP dataset [1] is a set of computer science bibliography. In our experiments, we used its subset DBLP AUTHORS ($72K$ citations) generated in [15], with publications in fields of databases, data mining and information retrieval. We regard each paper as a transaction and each author as an item. The correlation here describes the degree of the collaboration inside the group of authors. For $\theta = 0.007\%$ and $\gamma = 0.3$, NICoMiner is 20 times faster than the baseline method.
3. The COMMUNITIES dataset [12,13] is a publicly available dataset, which represents the demographic summarization for $1,980$ US communities. Each attribute value is a normalized numeric value between 0 and 1, which characterizes the relative presence of this attribute in a given community. We discretized each value into 5 equal-sized buckets: with ≤ 0.2 be very low and with > 0.8 be very high. Each community can be considered as a transaction, and each attribute-value pair as an item. The correlation here describes which demographic characteristics appear together in the same communities. COMMUNITIES is an example of a very dense dataset. The results in Figure 2(a) are for $\theta = 10\%$ and $\gamma = 0.60$. Even for this very high support threshold, the total number of frequent candidates exceeded the memory capacity of $40GB$, available in our experiments, and the results show the time before memory crashed: NICoMiner is more than 500 times faster than the baseline method. Note that using our new algorithm, we were able to lower the minimum support threshold for this dataset to 1% and obtain the results in just 12 seconds. This demonstrates the ability of NICoMiner to produce highly correlated patterns with low support, which for some datasets is even impossible using the frequency-based pruning alone.

In Table 6 we show some examples of patterns for each dataset, found among the top-20 correlations. These examples show that top correlations at low support can be used not only for such classic applications as product marketing, but also for the demographics analysis, or for the study of social networks.

For illustration, consider strong correlations extracted from the DBLP AUTHORS dataset (Figures 3(a)[1] and 3(b)[2]), where the edges label the degree of the pairwise correlation between authors. The nodes represent authors with 60 - 70 papers ($\theta = 0.001\%$). The pairwise correlations in Figures 3(a) and 3(b) are typical examples of (a) advisor-advisee relationships and (b) advanced mutual collaboration in an established collaborative group. Hence, such correlations can be used in studying evolving collaborations. Note that such strong correlations as in Figure 3(b) rarely take place in groups

[1] The letters in Figure 3(a) correspond to the following researchers: [A] *Hsinchun Chen*, [B] *Homa Atabakhsh*, [C] *Siddharth Kaza*, [D] *Jennifer Jie Xu*, [E] *Daniel Dajun Zeng*, [F] *Jialun Qin*, [G] *Yilu Zhou*, [H] *Chunju Tseng*.

[2] The letters in Figure 3(b) correspond to the following researchers: [K] *David A. Grossman*, [L] *Ophir Frieder*, [M] *Eric C. Jensen*, [N] *Steven M. Beitzel*, [O] *Abdur Chowdhury*.



(a) Pairwise correlations for typical advisor-advisee co-authorships.

(b) Pairwise correlations for advanced mutual co-authorships.

(c) Redundancy in (b) can be removed without information loss.

Fig. 3. Strong pairwise correlations in DBLP AUTHORS dataset

of authors with very high support. In general, for all datasets used in our experiments, the most interesting non-trivial correlations are found in the itemsets with low support.

Even though the number of correlated patterns is significantly smaller than the number of frequent itemsets, some of these patterns carry redundant information. As an extreme case, consider correlation value 1.00. The set of pairwise correlations in Figure 3(b) can be compressed without losing any information by replacing two authors M and N which co-authored in 100% of their papers by the joined item (MN). This removes significant amount of redundant correlations, as shown in Figure 3(c).

In addition, if the correlation values of the itemset and all its subsets are similar, they may be considered redundant. However in general, the correlation computed for a superset is not a redundant information, as can be shown on example in Figure 3(c). Based on values of pairwise correlations, we expect the correlation {K,M,N,O} to be at least as strong as {K,L,M,N}, while after computing actual correlations we find out that *Cosine*{*K,L,M,N*}$= 0.52$, while *Cosine*{*K,M,N,O*} is less than 0.1. This shows that information about mutual relationships of 3 or more objects cannot be deduced from pairwise correlations, and thus is not a redundant information. The distinction between redundant and non-redundant information represents the problem which requires special attention.

6 Related Work

The extension of association rules to correlations was introduced in the pioneering work of Brin et al. [3]. Since then, dozens of correlation measures have been proposed to assess the degree of the correlation. The comprehensive comparison of 21 different correlation measures can be found in [16], where the *null invariance* was introduced among other properties such as scaling-invariance and inversion-invariance. The importance of null-invariance for capturing meaningful correlations in large transactional databases was demonstrated later in [8,17,19]. In [19], the authors provide a unified definition of existing null-invariant correlation measures.

An efficient algorithm for correlation mining based on *All Confidence* and *Coherence* was proposed in [10,11]. In both papers, authors use the downward closure (or,

anti-monotonicity) property for pruning. In [19], authors derive an upper bound of *Kulczynski*, which was shown to be effective only for the comparatively high minimum support thresholds. The techniques based on sampling were recently proposed in [4], which are much faster, but at the cost of the incompleteness of results. Our approach works well for all null-invariant measures including $Kulczynski$ and $Cosine$, which did not have efficient algorithms for low support, and it produces the complete results.

Top-k correlated pattern mining was mostly developed only for 2-itemsets [22,23]. Our algorithm produces top-k correlations among itemsets with any number of items.

7 Conclusions and Future Work

In this paper, we addressed the problem of efficient mining of the top correlated patterns, based on any known null-invariant measure. We used *Cosine* correlation measure as an example, because it is one of the most widely-used, and at the same time, one of the most computationally challenging correlation measures. Even though it does not have the (anti)-monotonicity property, we developed two pruning methods that enabled an order of magnitude faster running time than the frequent pattern mining approach. We have shown experimentally that new pruning methods have high efficiency for discovering correlations in the itemsets with low support.

The top-k version of our new algorithm presents a valuable new tool to find top correlations. It can be easily extended to the problem of finding top-k correlations containing a particular item or pattern of interest (query pattern). This can be achived by maintaining a min heap data structure that keeps the top-k supersets of the query pattern.

In the future, we plan to address the problem of redundancy. If the correlation in the itemset is close to the correlation in its superset, it might be enough to output only the maximal superset pattern instead of reporting all patterns. One way to do it is to define a summary (or compressed) pattern for correlated patterns as in [20]. It would be interesting to incorporate the redundancy removal into the mining process, instead of performing it in a post-processing step.

Acknowledgement. The work was supported in part by the U.S. National Science Foundation grants IIS-09-05215 and OCI-07-25070, the Network Science Collaborative Technology Alliance Program (NS-CTA / INARC) of U.S. Army Research Lab (ARL) under the contract number W911NF-09-2-0053, and the Postdoctoral Fellowship of NSERC (Natural Science and Engineering Research Council) of Canada. Any opinions, findings, and conclusions expressed here are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Dataset: Dblp (2006), http://www.informatik.uni-trier.de/~ley/db/
2. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: SIGMOD (1993)
3. Brin, S., Motwani, R., Silverstein, C.: Beyond market baskets: generalizing association rules to correlations. In: SIGMOD (1997)

4. Campagna, A., Pagh, R.: Finding associations and computing similarity via biased pair sampling. In: Perner, P. (ed.) ICDM 2009. LNCS, vol. 5633, Springer, Heidelberg (2009)
5. Cohen, J., West, S.G., Cohen, P., Aiken, L.: Applied Multiple Regression Correlation Analysis for the Behavioral Sciences, 3rd edn. Lawrence Erlbaum Assoc Inc., Mahwah (2002)
6. Hahsler, M.: Groceries dataset (2007), http://rss.acs.unt.edu/Rdoc/library/arules/data/
7. Hahsler, M., Hornik, K., Reutterer, T.: Implications of probabilistic data modeling for mining association rules. In: Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation (2006)
8. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
9. Kuo, W.P., Jenssen, T.-K., Butte, A.J., Ohno-Machado, L., Kohane, I.S.: Analysis of matched mrna measurements from two different microarray technologies. Bioinformatics 18(3), 405–412 (2002)
10. Lee, Y.-K., Kim, W.-Y., Cai, Y.D., Han, J.: Comine: Efficient mining of correlated patterns. In: ICDM (2003)
11. Omiecinski, E.R.: Alternative interest measures for mining associations in databases. IEEE Trans. on Knowl. and Data Eng. 15, 57–69 (2003)
12. Redmond, M.: Communities and crime dataset (2009), http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime
13. Redmond, M.A., Baveja, A.: A data-driven software tool for enabling cooperative information sharing among police departments. European Journal of Operational Research 141, 660–678 (2002)
14. Srikant, R., Agrawal, R.: Mining generalized association rules. In: VLDB (1995)
15. Sun, Y., Han, J., Gao, J., Yu, Y.: iTopicModel: Information network-integrated topic modeling. In: Perner, P. (ed.) ICDM 2009. LNCS, vol. 5633, Springer, Heidelberg (2009)
16. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: KDD (2002)
17. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to Data Mining, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Amsterdam (2005)
18. von Storch, H., Zwiers, F.W.: Statistical analysis in climate research. Cambridge University Press, Cambridge (2002)
19. Wu, T., Chen, Y., Han, J.: Re-examination of interestingness measures in pattern mining: a unified framework. Data Min. Knowl. Discov. 21(3), 371–397 (2010)
20. Xin, D., Han, J., Yan, X., Cheng, H.: Mining compressed frequent-pattern sets. In: VLDB (2005)
21. Xiong, H., He, X., Ding, C.H.Q., Zhang, Y., Kumar, V., Holbrook, S.R.: Identification of functional modules in protein complexes via hyperclique pattern discovery. In: Pacific Symposium on Biocomputing (2005)
22. Xiong, H., Zhou, W., Brodie, M., Ma, S.: Top-k ϕ correlation computation. INFORMS Journal on Computing 20(4), 539–552 (2008)
23. Zhu, S., Wu, J., Xiong, H., Xia, G.: Scaling up top-k cosine similarity search. Data Knowl. Eng. 70, 60–83 (2011)

Smooth Receiver Operating Characteristics (*smROC*) Curves

William Klement[1], Peter Flach[2], Nathalie Japkowicz[1], and Stan Matwin[1,3]

[1] School of Electrical Engineering and Computer Science, University of Ottawa, Canada
{klement,stan,nat}@site.uottawa.ca
[2] Computer Science, Bristol University, BS8 1UB United Kingdom
Peter.Flach@bristol.ac.uk
[3] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland

Abstract. Supervised learning algorithms perform common tasks including classification, ranking, scoring, and probability estimation. We investigate how scoring information, often produced by these models, is utilized by an evaluation measure. The ROC curve represents a visualization of the ranking performance of classifiers. However, they ignore the scores which can be quite informative. While this ignored information is less precise than that given by probabilities, it is much more detailed than that conveyed by ranking. This paper presents a novel method to weight the ROC curve by these scores. We call it the Smooth ROC (*smROC*) curve, and we demonstrate how it can be used to visualize the performance of learning models. We report experimental results to show that the *smROC* is appropriate for measuring performance similarities and differences between learning models, and is more sensitive to performance characteristics than the standard ROC curve.

1 Introduction

Supervised learning algorithms perform common learning tasks including classification, ranking, scoring, and probability estimation. This paper investigates how scoring information, often produced by such algorithms, may be utilized by the performance evaluation measure. A scoring model estimates scores on the training data and assigns them to testing data to express class memberships, and sometimes, these may represent probabilities where Brier Score is used to assess their quality. However, when the scores are not probabilities, they are more than ranks; they aren't just ordinals but numbers expressed on some scale. The scale may be unbounded and may not be additive, eg. the score could be a likelihood ratio which is multiplicative. Many applications, particularly in medicine, employ scores that are highly meaningful for the users and are not probabilities, eg. ICU scoring systems. In these cases, the task is usually reduced to a ranking or a classification by ignoring the magnitudes of scores.

The Receiver Operating Characteristics (ROC) curves are commonly used to visualize the ranking performancs of classifiers. However, they ignore the

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 193–208, 2011.

scores which, we argue, are quite informative. For instance, the scores convey information as to how close two data points may be from one another within a given rank. While such information is less precise than that of probabilities, it is much more detailed than performance information conveyed by ranking.

To illustrate this concept, consider the problem of assessing similarities and differences among user preferences. For example, Anna and Jan are asked to make movie recommendations based on their preferences. They are both given the same list of n movies to which they assign a *positive*, or a *negative* recommendation along with a continuous score (between zero and one). The score indicates the degree to which they like or dislike the movie, with value 1 indicating the maximum *"liking"* and 0 the maximum *"disliking"*. The task is to examine recommendations and preferences made by Anna and by Jan in search for similarities and differences between their assessments of the movie collection. The consideration of both criteria, recommendations and preferences, makes this task considerably more complex. For instance, Anna positively recommends a movie with a score of 0.52 because although she does not like the movie per se, she finds this movie worthy of recommendation. However, Jan gives the same movie a negative recommendation but assigns to it a score of 0.6, because while he likes it, he does not find it worthy of recommendation. Clearly, if we only compare their recommendations, we may draw a conclusive disagreement. Similarly, their scores depict a disagreement in the opposite direction. The issue becomes: do they really disagree? Her lower score suggests agreement with Jan's decision of a *not-so-good* movie, but his high score indicates his inclination to a *not-so-bad* movie. Although their assessments may appear to disagree, there is, in fact, a substantial agreement among them. This agreement, or the lack there of, between Anna and Jan is expressed as a combination of both a binary decision and a continuous score, which is far from being probabilistic in nature.

To compare Anna's results to Jan's we would plot and compare two ROC curves (or two *smROC* curves for that matter). The standard ROC method only compares positive to negative recommendations. The AUC, in this case, depicts how they agree on separating the positive from the negative recommendations only because the magnitude of their preference scores are excluded from the analysis. The proposed method addresses this issue of assessing both decisions and scores using a single evaluation measure based on ROC analysis. The area under the proposed curve (the *smAUC*) will depict the separation of scores in the context of the binary decision. This problem reaches several domains including: search engines where the results of a query may (or may not) be relevant and are strongly (or weakly) related to a query, recommendation systems as illustrated in the above example, medical decision making where a physician is interested in the presence (or absence) of a condition along with its associated severity score, and finally, in bioinformatics where a genetic sample may be analyzed for up (or down) regulation of a particular gene at a high (or low) levels of gene expression.

The *smROC* is a novel method that extends the ROC curve to include the scores as smoothing weights added to its line segments. We, therefore, call it the Smooth ROC (*smROC*) curve. This proposed *smROC* method measures

similarities and differences in how Anna and Jan make recommendation decisions and assign scores and can be used to compare scoring classifiers, but this is hardly the only application. The *smROC* method can also be used to measure similarities and differences among data points. Optimizing such an agreement, or the lack thereof, will offer substantially more informative views of various abilities of supervised machine learning methods.

The organization of this paper follows the presentation of Section 2 to motivate the research and to discuss related work, Section 3 describes how to construct *smROC* curves and shows calculations of *smAUC*, and Section 4 presents experimental results that demonstrate the superiority of *smROC* over the standard ROC in measuring performance similarities and differences among scoring classifiers. Finally, Section 5 concludes with a brief discussion of future work.

2 Motivation and Related Work

Common supervised learning tasks (Figure 1) convey diverse information of class memberships of data points. Classification is a categorization of points into classes with yes/no decisions. Binary classification is a special case of making decisions on two classes. Ordinal classification extends the multi-class settings and imposes an order on the classes. However, its outcome remains a classification. Ranking yields an order of data points, and intuitively, a good ranker places the positives towards the top and the negatives towards the bottom of an ordered list. Ranking can be depicted by a simple order or by assigning ranks to data points, the latter can also be viewed as scores. Scoring enables the learning model to convey its confidence in class memberships [7]. However, interpreting scores, in general, requires a considerable amount of information related to the definition of the underlying scoring function, which is usually difficult to obtain and is highly uncertain. Instead, some algorithms estimate class memberships probabilities to induce this information. Thus, a probability estimation task may be considered an informed case of a scoring task that involves modeling the posterior probability distribution of class memberships given the training data.

The ability to distinguish between data points depends on the granularity of predictions made by the model and is known as the "discriminancy" of a

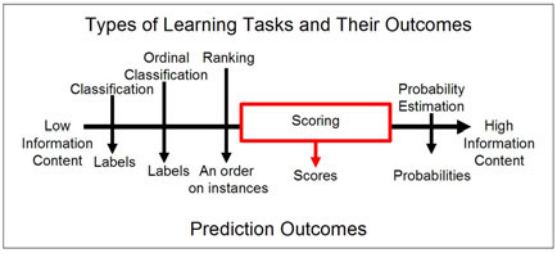
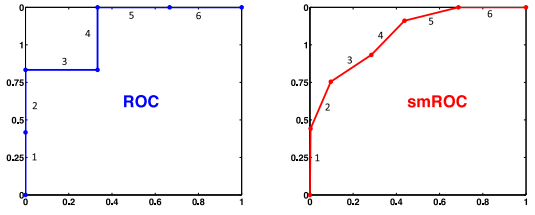


Fig. 1. Information content of common machine learning tasks

performance measure [11] where ties between predictions represent a difficult challenge. While classification distinguishes between positives and negatives, ranking separates instances based on how high (or low) they are placed in an ordered list. Yet, gaps between the ranked instances are determined by their availability rather than by the magnitudes of class memberships usually conveyed by the scores. Obviously, a prediction conveyed by a continuous-value score is less likely to produce ties than a binary decision. The latter is the least informative decision whose outcomes offer the least granularity. Probability estimation is the most informative task designed to describe the probability distribution learned from data. Its wealth of information offers a high degree of granularity due to exploiting powerful statistical principles. These methods, we find, are restrictive, inflexible, and parametric in nature. For instance, most machine learning methods assume the class distribution invariant between training and testing [8], and the quality of probability estimation is affected by this assumption [2]. Furthermore, the naive Bayesian learning method assumes the independence of attributes that describe the data. In practice, such assumptions are often violated [2,9,10], and in many cases, the raw scores provide poor estimates of true probabilities because some models are prone to estimating poor probabilities [5]. Or, it may be unnecessary to treat the scores as calibrated posterior probabilities. In these cases, the scores are merely used to construct the ROC curve for the purpose of performance analysis [5]. This use of the scores represents a reduction in information conveyed by the model (Figure 1) and we argue that it omits information relevant to the performance analysis by eliminating the magnitudes of the scores. The *smROC* curve evaluates such scores, particularly when their estimates are poor or violate assumptions required by probabilistic methods.

The scope of performance measures is usually restricted by information made available by the model. Scalar evaluation metrics such as accuracy, precision, recall, AUC, and MSE (or Brier score [3]) compute a summary of performance insensitive to characteristics of class memberships of individual points. The ROC method addresses this issue but ignores the magnitudes of the scores. Their exclusion can result in plotting identical ROC curves for multiple learning models irrespective of differences in their score assignments. Wu et al. [15] propose the sAUC method to incorporate score margins into the standard AUC and to detect the existence of fixed-size score gaps between positive and negative data points. Effectively, this measures how quickly the AUC, under the standard ROC curve, deteriorates when the positive scores are decreased. The *smAUC* method differs from sAUC in several ways. While *smAUC* weights the standard ROC curve by the absolute values of the scores, sAUC relies on score differences being greater than a fixed gap. In addition, the sAUC measures these score gaps only between positive and negative data points, which limits the assessment to the positives versus the negatives [15]. In contrast, the *smROC* relies on the score values themselves which enables the visualization of score differences and gaps of any size between any pairs of points regardless of their class, i.e., the *smAUC* can distinguish between data points in one class. Going back to our example, if Anna positively recommends two movies at scores 0.7 and 0.99, the sAUC will not



Anna's Assessment

i	*Decision*	*Score*
1.	+	0.99
2.	+	0.70
3.	-	0.60
4.	+	0.51
5.	-	0.20
6.	-	0.00

Fig. 2. In the ROC space, it's difficult to distinguish between movies 1 and 2. Anna recommends both but she likes movie 1 almost 30% more than 2. The *smROC* plots the line segments with slopes proportional to the scores. Visually, movies 1 and 2 have different slopes in the *smROC* space. Similarly, scores of movies 5 and 6 result in different slopes. Anna likes movie 1 the most and 6 the least.

compare these two movies because they are both positives. The *smROC* curve represents individual movies by line segments whose slope are proportional to the corresponding score value (Figure 2). Therefore, the area under this curve will be affected by the exact margin of these two points. If we examine the definition of the *smAUC* (in the next section), it is clear that the *smAUC* compares all pairs of data points, and thus, these two movies will contribute their score magnitudes to the *smAUC*. This means that the *smAUC* reports a different kind of performance information than the sAUC. The *smAUC* metric depicts the performance of ranking weighted by the magnitude of confidence in predictions (conveyed by the class membership scores). The sAUC relates to the behavior of the AUC, under the ROC, when the positive scores are decreased.

A recent study [14] argues that soft variations of the AUC metric contribute little to the AUC for the purpose of model selection, because they favor models that generate large (rather than small) score margins. In this paper, we argue that such *soft* analysis of the ROC can be used to understand the behavior of scores. The *smROC* method not only produces a visualization of the scores (as opposed to their margins), it also detects similarities among score values. Such analyses are not limited to model selection, they can help compare scores assigned to data points as well. This aspect of performance cannot be measured by the standard AUC metric.

3 Constructing a Smooth ROC Curve

Let X be a data set that contains $n = n^+ + n^-$ (positive and negative) points, and let x_i be the i^{th} point whose label is $C_i \in \{+, -\}$ and S_i be the positive class membership score assigned to x_i. Finally, let m^+ and m^- be the average positive and negative scores respectively. Algorithm 1 begins at the origin and incrementally plots the ROC curve by examining points in X in a decreasing order of their S_i scores [5] (Figure 2). For a positive x_i, the curve climbs one step upwards, and for a negative x_i, the curve runs one step to the right. The vertical

Algorithm 1. Incrementally plotting of the ROC curve [5]

Input: $n = n^+ + n^-$ positive and negative points, $S_i \in [0,1]$ scores of n points in a decreasing order, and $C_i \in \{+,-\}$ Labels.
for $i = 1$ **to** n **do** {start at the origin (0,0)}
  if scores are tied between y positives and x negatives **then**
    simultaneously move up by $\frac{y}{n^+}$ and right by $\frac{x}{n^-}$
  else if $C_i = +$ **then**
    move up by $\frac{1}{n^+}$
  else if $C_i = -$ **then**
    move right by $\frac{1}{n^-}$
  end if
end for

and horizontal step sizes are $\frac{1}{n^+}$ and $\frac{1}{n^-}$ in the ROC space. To incorporate S_i scores into this curve, we modify Algorithm 1 to produce Algorithm 2. Our approach relies on altering the step sizes in the space proportionally to the score magnitudes. This modification preserves properties and characteristics of the ROC curve so that ROC analysis remain valid.

A scoring classifier estimates class membership scores and assigns them to data points. Classifications are obtained by imposing a threshold on these scores. While negative points whose scores lie above the threshold result in false positive errors, positives with scores below that produce false negative errors. We argue that such errors can be blamed not only on the choice of threshold value, but also on the *inappropriateness* of scores (see definition 1). More specifically, *inappropriate* scores result from assigning low scores to positives and/or from assigning high scores to negatives. The identification of high and low scores relies on a midpoint value in the range of the scores. When the scores are calibrated, this midpoint lies naturally at 0.5. Otherwise, it can be estimated. The latter is discussed in details later in this section.

Definition 1. *For a midpoint Mid, a class membership score S_i is appropriate for data point x_i if S_i is greater or equal to Mid when x_i is positive, or if S_i is less than Mid when x_i is negative. Otherwise, $(1 - S_i)$ is appropriate for x_i.*

To deal with score appropriateness, two separate confusion matrices are needed as shown in Figure 3. The separate consideration of classification accuracy for points whose scores are appropriate or inappropriate depicts the ability of the model to counter score inappropriateness. Algorithm 2 treats appropriate scores differently from inappropriate ones. For appropriate scores, it climbs vertically proportionally to the magnitude S_i while running horizontally proportionally to $(1 - S_i)$. (the normalization factors $\propto_h$ and $\propto_v$ are omitted at this point for simplicity). The first plot from the left in Figure 3 shows a positive data point with an appropriate score ($S_i > Mid$ assuming $Mid = 0.5$). This is depicted by a higher rise than run. When $S_i < Mid$, x_i is more likely to be negative (the second plot from the left in the same figure), and the score is deemed appropriate when the $C_i = -$ because S_i agrees with the C_i. While S_i indicates,

(Score Appropriateness)

Label	Scores: High	Low
+	yes	no
−	no	yes

(Accuracy of Appropriate Scores)

Score	Label	Predicted: Y	N
High	+	correct	incorrect
Low	−	incorrect	correct

(Accuracy of Inappropriate Scores)

Score	Label	Predicted: Y	N
High	−	incorrect	correct
Low	+	correct	incorrect

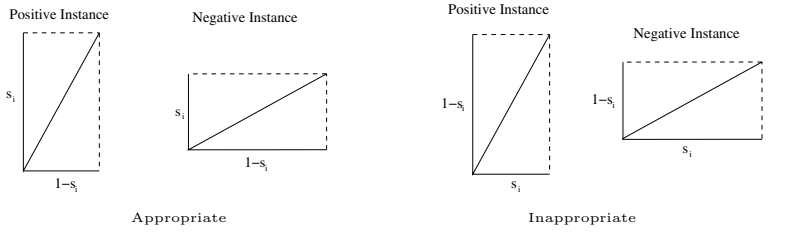


Fig. 3. The modified step size is based on score appropriateness

in the appropriate case, whether the x_i is positive or negative, $1 - S_i$ contradicts the label $C_i \in \{0, 1\}$. Thus, we plot a vertical climb proportional to S_i to show a gain in performance, and impose a horizontal penalty proportional to $1 - S_i$. Inappropriate scores must be treated differently. For instance, a positive point may be assigned a low score, and/or a negative point may be assigned a high score. These two situations are considered inappropriate, which we account for by reversing the score plotting strategy as shown in the two plots on the right of Figure 3. A positive point that is assigned $S_i < Mid$ conveys that x_i is more likely to be negative and contradicts the positive label. Similarly, a negative x_i that is assigned ($S_i > Mid$) suggests that x_i is more likely to be positive. In both cases, S_i contradicts the label and $1-S_i$ agrees with the label. Such contradictions merit adjustments. Therefore, Algorithm 2 plots a performance gain in the form of a vertical climb and proportional to $1-S_i$, and it impose a penalty as a horizontal run proportional to S_i. Finally, the complete curve is assembled by connecting individual vectors resulting from the successive assessments of individual points similar to Algorithm 1.

The main difference between the two algorithms lies in the adjustment of the step-size. In the standard ROC curve (Algorithm 1) individual line segments either rise by $\frac{1}{n^+}$ or run by $\frac{1}{n^-}$ but not both. In the *smROC*, progress is made in both directions simultaneously using the scores (as described above) and using $\propto_v$ and $\propto_h$ as the vertical and horizontal normalization factors respectively. Algorithm 2 climbs vertically by $\frac{S_i}{\propto_v}$ while simultaneously running horizontally by $\frac{(1-S_i)}{\propto_h}$ when the scores are appropriate, and when they aren't, this algorithm climbs up by $\frac{(1-S_i)}{\propto_v}$ and simultaneously runs by $\frac{S_i}{\propto_h}$. Such a curve is illustrated in Figure 2. The remaining issues include the calculations of Mid, $\propto_v$, and $\propto_h$.

The midpoint Mid is necessary for the assessment of score appropriateness to separate high from low scores. When the scores $S_i \in [0, 1]$ are calibrated, it is natural to use $Mid = \frac{max(S)-min(S)}{2} = \frac{(1-0)}{2} = 0.5$. However, when the scores are uncalibrated, we estimate the midpoint between the average positive score

Algorithm 2. Incrementally Plotting the *smROC* curve

1: **Input:** $n = n^+ + n^-$ positive and negative points, $S_i \in [0,1]$ scores of n points in a decreasing order, $C_i \in \{+,-\}$ Labels, Mid = Equation 1, $\propto_v$ = Equation 2, and $\propto_h$ = Equation 4.
2: **for** $i = 1$ to n **do** {start at the origin (0,0)}
3: **if** scores are tied (above Mid) between y positives and x negatives **then**
4: Move up $\frac{yS_i + x(1-S_i)}{\propto_v}$ and move right $\frac{y(1-S_i) + xS_i}{\propto_h}$
5: **else if** scores are tied (below Mid) between y positives and x negatives **then**
6: Move up $\frac{y(1-S_i) + xS_i}{\propto_v}$ and move right $\frac{yS_i + x(1-S_i)}{\propto_h}$
7: **else if** $(C_i = +)\text{AND}(S_i > Mid)$ **then**
8: Move up $\frac{S_i}{\propto_v}$ and move right $\frac{(1-S_i)}{\propto_h}$
9: **else if** $(C_i = -)\text{AND}(S_i < Mid)$ **then**
10: Move up $\frac{S_i}{\propto_v}$ and move right $\frac{(1-S_i)}{\propto_h}$
11: **else if** $(C_i = +)\text{AND}(S_i < Mid)$ **then**
12: Move up $\frac{(1-S_i)}{\propto_v}$ and move right by $\frac{S_i}{\propto_h}$
13: **else if** $(C_i = -)\text{AND}(S_i > Mid)$ **then**
14: Move up $\frac{(1-S_i)}{\propto_v}$ and move right by $\frac{S_i}{\propto_h}$
15: **end if**
16: **end for**

m^+ and the average negative score m^- for a given data set X. When the scores S_i are calibrated, and if the class distribution $c = \frac{n^+}{n^-} = 1$, then, it can be shown that $m^+ + \frac{m^-}{c} = 1$. This becomes obvious when $S_i \in \{0,1\}$ and $c = 1$ which gives $n^+m^+ + n^-m^- = n^+$. However, in the general case where scores S_i are not calibrated and $c \neq 1$, $m^+ + \frac{m^-}{c}$, reduces to $\frac{sum(S)}{n^+}$. Therefore, we set Mid to the midpoint as per Equation 1. This estimation of Mid is data specific and is based on scores produced by the model. Alternate methods of estimating Mid remain under investigation.

$$Mid = \frac{1}{2}(m^+ + \frac{m^-}{c}) = \frac{sum(S)}{2n^+} \tag{1}$$

To ensure that Algorithm 2 makes progress vertically and horizontally in the unit square of the space, the step-size must be normalized. These vertical and horizontal normalization factors are represented by $\propto_v$ and $\propto_h$ respectively. We now show their calculations using H^+, L^-, L^+, and H^- as the sets of: positives where $S_i \geq Mid$, negatives with $S_i < Mid$, negatives of $S_i \geq Mid$, and positives where $S_i < Mid$ respectively. Algorithm 2 plots the curve in steps proportional to S_i or $1 - S_i$ scores. We divide each step by the total score contributions in either direction to normalize them. To determine the vertical normalization factor $\propto_v$, we add up scores contributing to the upwards progress. Lines 7 and 9 of Algorithm 2 show this for points in H^+, and in L^- respectively, they contribute their S_i scores towards a vertical climb upwards. Lines 11 and 13 show that points in L^+, and in H^- respectively, contribute their $1 - S_i$ scores also in the

vertical direction. Therefore, this total sum of scores contributions in the vertical direction is computed by Equation 2 as the sum of the positive contributions $\Theta(x_i)$ for $i = 1 \ldots n$.

$$\propto_v = \sum_{i=1}^{|H^+|} S_i \; + \sum_{i=1}^{|L^-|} S_i \; + \sum_{i=1}^{|L^+|} (1 - S_i) \; + \sum_{i=1}^{|H^-|} (1 - S_i) = \sum_{i=1}^{n} \Theta(x_i) \tag{2}$$

$$\Theta(x_i) = \begin{cases} s_i & \text{if } x_i \in \{H^+ \cup L^-\} \text{ (Appropriate Scores)} \\ 1 - s_i & \text{if } x_i \in \{H^- \cup L^+\} \text{ (Inappropriate Scores)} \end{cases} \tag{3}$$

The horizontal normalization factor $\propto_h$ is the sum of all scores contributions towards shifts (to the right) in the horizontal direction. Instances in H^+, and in L^-, respectively on Lines 7 and 9 of Algorithm 2, contribute their $(1 - S_i)$ scores along the horizontal direction. In addition, points in L^+, and in H^-, respectively on Lines 11 and 13 of Algorithm 2, contribute their S_i scores also to the horizontal progression. Therefore, the horizontal normalization factor $\propto_h$ is computed by Equation 4 as the sum of the negative contributions $1 - \Theta(x_i)$ for $i = 1 \ldots n$.

$$\propto_h = \sum_{i=1}^{|H^+|} (1 - S_i) \; + \sum_{i=1}^{|L^-|} (1 - S_i) \; + \sum_{i=1}^{|L^+|} S_i \; + \sum_{i=1}^{|H^-|} S_i = \sum_{i=1}^{n} (1 - \Theta(x_i)) \tag{4}$$

The area under the *smROC* curve (*smAUC*) is calculated using Equation 5. The *smAUC* is based on accumulating the product of positive score contributions ($\Theta(x_i)$) by the negative score contributions for all data points ranked lower than x_i. The latter is computed by Equation 6. A special case occurs when x_i is compared to itself, only one half of the product contributes toward the area under the curve in the second case of Equation 6.

$$smAUC = \frac{1}{\propto_v \propto_h} \sum_{i=1}^{n} \sum_{j=1}^{n} \Theta(x_i) \Psi(x_i, x_j) \tag{5}$$

where:

$$\Psi(x_i, x_j) = \begin{cases} 1 - \Theta(x_i) & \text{for } (S_i > S_j) \text{ and } (i \neq j) \\ \frac{1}{2}(1 - \Theta(x_i)) & \text{for } i = j \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

The *smAUC* represents the separation between the total positive contribution of scores $\Theta(x_i)$ and the total negative contribution of scores $1 - \Theta(x_i)$ for all instances $i = 1 \ldots n$ in their ranking order ($S_i > S_j$). This suggests that *smAUC* favors scores that result in higher separation of classes weighted by the magnitudes of the scores. Finally, it can be shown that when the scores S_i are zeros and ones, the *smROC* and the *smAUC* will reduce to the standard ROC and the standard AUC respectively. This discussion is omitted due to space limitations.

Table 1. UCI binary classification data [1]

Abbr.	Data Set Name	n	$\|+\|$	$\|-\|$	Features	%+
prom	promoters	106	53	53	57	50
echo	echocardiogram	132	43	88	7	33
hepa	hepatitis	155	32	123	19	21
prks	parkinsons	195	147	48	22	75
hart	statlog heart	270	120	150	13	44
hrth	heart disease hungarian	294	188	106	13	64
hors	horse-colic reduced	296	188	108	21	64
habr	haberman	306	81	225	3	26
iono	ionosphere	351	225	126	34	64
vots	house-votes-84	435	168	267	16	39
jcrx	japanese crx	690	307	383	15	44
aust	statlog australian	690	307	383	14	44
wisc	breast cancer wisc	699	241	458	9	34
blod	blood transfusions	748	178	570	4	24
diab	pima-indians-diabetes	768	268	500	8	35
mamo	mammographic masses	945	434	511	5	46
tic	tic-tac-toe	958	626	332	9	65
ger	statlog german	1000	700	300	20	70
oz8h	ozone eighthr	2534	160	2374	72	6
oz1h	ozone onehr	2536	73	2463	72	3
chss	chess kr-vs-kp	3196	1669	1527	36	52
ads	internet ad	3279	459	2820	1558	14
spam	spambase	4601	1813	2788	57	39
mush	mushroom	8124	3916	4208	23	48
mgic	magic04	19020	12332	6688	10	65
adlt	census adult	32562	7841	24720	14	24

4 Experiments

The objective of this experiment is to illustrate the ability of the *smROC* curve to measure performance similarities and differences of scoring classifiers, and to demonstrate its superiority over the standard ROC curve. Therefore, we first construct learning models that are expected to produce similar performance, and we assess their performance in the *smROC* space and in the standard ROC. We wish to show that the *smROC* methods captures their performance similarities with higher performance sensitivity than the standard ROC. Then second, we generate performance data for two learning methods, Naive Bayes (NB) and Probability Estimating Trees (PET – unpruned decision trees with Laplace correction [12]), which are known to produce different scores when applied to the same data. Again, we compare their performance analysis in the *smROC* space and in the standard ROC space to demonstrate that the *smROC* method captures score differences better than the standard ROC.

To simulate score similarities, we rely on the consistency of the learning method by fixing the learning algorithm, as well as, the training/testing data

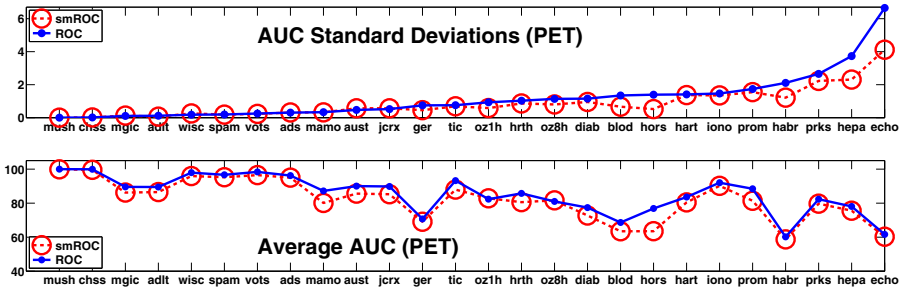


Fig. 4. Similar PET Models from ten runs of 10-fold cross-validation

distribution. The idea is to construct, more-or-less, similar classifiers from randomized versions of the same data. For instance, a collection of ten Naive Bayes classifiers should produce similar class membership scores in multiple runs of 10-fold cross-validation applied to the same data. In this case, performance variations occur due to the random splitting of data into 10 folds. The same argument applies to building ten PETs in the same way. We use the ROC and the *smROC* analysis to show that classifiers in the same group produce similar scores. As for differences, we rely on the same two methods, PET and Naive Bayes, to produce significantly different scores from each other as evidence suggests [12,13,6,16]. Therefore, to measure performance differences, we compare the pairwise performance of PET and Naive Bayes over the multiple runs of 10-fold cross-validation, they should produce different performance. We then compare how well the standard ROC curve and the *smROC* curve capture these differences. An issue we need to consider is the construction the pairs of models which should be derived from identical data sets in each run. This accomplished by controlling the seeds when we randomly split the data into folds.

For the purpose of this experiment, we use benchmark data sets listed in Table 1, which are obtained from the machine learning repository [1]. For evaluation, we construct the *smROC* and the standard ROC curves (two curves) for the two learning methods resulting from each of the ten runs on every data set (twenty-six sets). This generates over one thousand curves for us to analyze. To make this analysis manageable, we summarize the curves by their respective area under the curve. Thus, we calculate the *smAUC* and AUC respectively. And for ease of presentation, we plot the average and standard deviation of AUC and of *smAUC* for each data set. When assessing similarities. we plot these values for each model separately, but for differences, we plot these values (average and standard deviation) for their observed pairwise difference.

4.1 Performance Similarities

Figure 4 shows the standard deviation and the average area under the curves (*smROC* and ROC) generated by the PET method over ten runs of 10-fold cross-validation. We observe that the dashed curve is consistently below (or sometimes

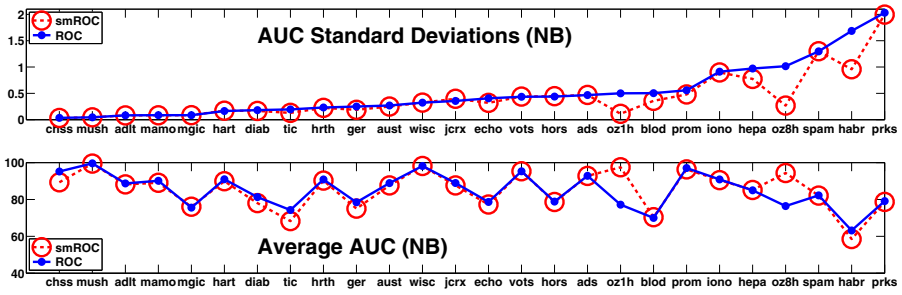


Fig. 5. Similar Naive Bayes models from ten runs of 10-fold cross-validation

the same) as the solid curve in both plots. For data sets that produce higher standard deviation (in the top plot), the standard deviation of the area under the *smROC* curve is lower than that under the standard ROC. When this standard deviation is low, both ROC and *smROC* curves show the same low standard deviation. This suggests that when variations occur among similar models, the *smROC* captures more similarity than the standard ROC. The higher standard deviation of area under the ROC curve can be attributed to the exclusion of score magnitudes, which results in an over/under estimate of separation between ranks (in the ROC space, the area under the curve is estimated by a discrete indicator function, whereas, the *smAUC* follows the separation between score values). This observation is supported by the bottom plot (in the same figure) which shows the average area under the curve. The average area under the *smROC* curve is generally lower than that under the standard ROC in the figure. This makes sense in the context of the *smROC* being a kind of smoothing of the ROC curve (weighted by score magnitudes). The PET learning method uses unpruned decision trees with Laplace correction for smoothing the scores. Thus, we expect its associated *smROC* curves to be smoother than the corresponding ROC curves. In a sense, the consistently lower *smAUC* is saying that PET scores are consistently smoother than what the standard ROC shows.

Similar observations can be made in Figure 5. For data sets with low standard deviation, the *smROC* and the standard ROC produce similar standard deviations of the area under their respective curves for similar Naive Bayes models. As this standard deviation increases, the *smAUC* produces lower standard deviation (in the top plot of the figure). The average area under the *smROC* curve is the same, or less, than that under the standard ROC curve for most data sets. One exception is the `oz1h` and `oz8h` data sets (they are both obtained form the same *Ozone* domain [1]). For these sets, the average *smAUC* is substantially higher than the standard AUC. Furthermore, the standard deviation of the area under the *smROC* curve for these two sets is also substantially lower than that under the standard ROC curve (see the top plot of the same figure, the corresponding open circles are much lower than the solid curve). This suggests that the ranking order of data points fails to correctly separate the two classes.

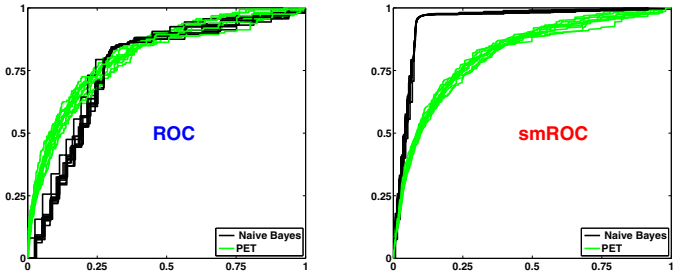


Fig. 6. Naive Bayes and PET models constructed from `oz8h` data

However, the combination of score magnitudes along with the ranking performance produce better class separation. Perhaps, the standard ROC curve is faced with classification errors due to the scores being just above, or just below, the classification threshold. Consequently, the scores appear in disagreement with class labels (this is similar to the movie recommendation example presented in the introduction). Such errors are compensated for when the magnitudes of the scores are used as weights by the *smROC* analysis.

4.2 Detecting Differences

In this section, we compare the performance of two learning methods, Naive Bayes and PET, which are known to be different in how they produce class membership scores. We measure their performance on our data sets in both the *smROC* and the standard ROC spaces. Figure 6 presents the ROC and the *smROC* curves for the two learning methods applied to the `oz8h` data. Comparing the ROC curves leads to the conclusion that both methods accomplish comparable performance because Naive Bayes (the dark curves) and PET (the light curves) are close to each other. In addition, if we compare the average area under their curves respectively, we measure a small difference between them, well, smaller difference than that observed in the *smROC* space. The reason the two plots differ is due to including score magnitudes in the construction of the *smROC* curves. The left plot of Figure 6 shows that the ROC curve is insensitive to differences in scores. All we can see is that the PET curves are visually smoother than those of the Naive Bayes'. However, if we use the AUC metric, this difference becomes far less obvious. Alternatively, the *smROC* curves show that scores produced by Naive Bayes are high for positive examples (the steep vertical rise) and they are low for negative instances (the consistent horizontal run in the top right). The strong change of direction along the *smROC* curves associated with naive Bayes indicates a substantial gap in the scores. The *smROC* curves associated with the PET models (the light curves) appear smoother with a comparable area under the curve in both spaces. This illustrates how the *smROC* curve depicts ranking information but adds score magnitudes. These magnitudes have a little smoothing effect on the standard ROC in the case of PETs.

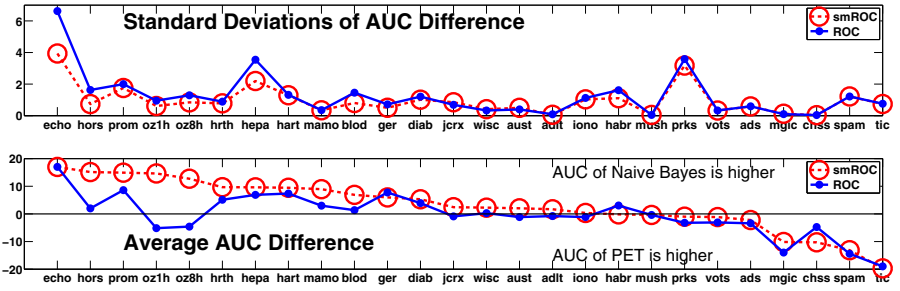


Fig. 7. AUC difference for Naive Bayes and PET models

However, in the case of Naive Bayes, the score magnitudes amplify its classification ability but expose the *definite* nature of its scores (definite, or appropriate, because positive examples have high scores and negatives have low scores). This highlights a significant difference between the two learning methods that the standard ROC shows little to no sensitivity to. In fact, if we rely on the area under the curve metric to understand such differences, the *smAUC* is more favorable due to its sensitivity to the scores. The standard AUC fails to measure differences in scores which presents an interesting argument for studies such as [14]. Vanderlooy and Hüllermeier [14] suggest that soft variations of the AUC offer little to no improvement when it comes to model selection. This may well be the case when the primary interest is classification or ranking for that matter. However, when it comes to examining a scoring method where the scores are of interest, our results show that the *smAUC*, a soft variation of the standard AUC, is able to measure scoring differences that are buried in the ROC space due to the exclusion of their magnitudes. This paper argues and shows that these differences are important. For instance, the scoring behavior of Naive Bayes and decision trees have been the focus of several studies [12,13,6,16]. Our *smROC* curve depicts these differences with ease, and moreover, the *smAUC* represents a metric sensitive to such characteristics. If the *smAUC* metric enables these studies to investigate the scoring behavior of learning methods with ease.

Lets consider the average and standard deviations of the difference in the AUC and in the *smAUC*, respectively, for our two learning methods. Figure 7 shows the average pairwise difference of the area under the curve between Naive Bayes and PET resulting from this experiment. We now describe how we compute this difference; in each of the ten runs of 10-fold cross-validation, we construct a Naive Bayes model and a PET model from the same sample of data (randomly split into folds using the same seed). Then, we construct their ROC and *smROC* curves, and we compute their respective areas under the curves and record their difference. Namely, we subtract the area covered by the curve associated with PET from the area covered by the curve associated with Naive Bayes. At the end, we compute the average and the standard deviation of these recorded differences in their respective spaces. Finally, we plot these results for all data sets (see Figure 7). It is clear that the average AUC difference in both spaces agree when

this difference is in favor of the PET models (the two curves agree when they are close or below the solid line of 0 difference in the right half of the bottom plot of the Figure 7). In addition, if we examine the standard deviations for the same data sets in the top plot, we see that both AUC and *smAUC* produce similar standard deviations of this difference. This suggests that PET models that perform better than Naive Bayes produce solid, and consistent, ranking and scoring performance observed in both spaces. However, when the balance tips in favor of the Naive Bayes models, the scores become more appropriate (as we define them). The *smAUC* measures a substantially higher difference than the standard AUC. This is illustrated by the average difference in the ROC space being below that of the *smROC* curve, and sometimes, the former crosses below the zero line (see the bottom plot of the same figure). Furthermore, the observed standard deviation of this AUC difference is higher for ROC curves than for *smROC* curves (see the corresponding standard deviations in the top plot of the same figure). This suggests that the ROC curve struggles to measure this difference between the two models. Thus, the use of the standard AUC fails to detect these differences because they are excluded. The *smAUC*, however, measures these differences clearly and with lower standard deviations. Since it favors appropriate scores, these results suggest that Naive Bayes produces scores useful for classification but they are far from being smooth.

5 Conclusions

This paper presents a novel evaluation measure, the *smROC* curve, to incorporate class membership scores into the ROC curve. Based on a categorization of common machine learning tasks, which include classification, ranking, scoring and probability estimation, we argue that class membership scores convey valuable information relevant to the performance. Ignoring them, as the standard ROC does, results in a reduction of information expressed by the model.

Our results show that the *smROC* is effective in measuring performance similarities and differences among learning models. The *smROC* is sensitive to performance characteristics related to how a learning model assigns class membership scores to data points. The results demonstrate that the *smROC* curve measures the performance with less variations than the standard ROC curve, and it detects performance differences more consistently than the standard ROC. These results are statistically significant using the paired t-test. However, significance results are omitted due to space limits. Therefore, the *smROC* method enhances the ROC method, and captures specialized performance information with a higher granularity while remaining more abstract than dealing with probability estimates. Future research directions include investigating alternate methods of computing the mid point used to assess score appropriateness, analyzing the effect of varying this midpoint for all values between zero and one, and exploring other advantages of the *smROC* curve. These include its sensitivity to changes in the domain, i.e., it can be shown that the *smROC* method is sensitive to changes in the data distribution. It can be argued that measuring these differences between training and testing represents a significant accomplishment.

Acknowledgement. This research has been supported by the Natural Sciences and Engineering Research Council of Canada and the Ontario Centres of Excellence.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
2. Bennett, P.N.: Using Asymmetric Distributions to Improve Text Classifier Probability Estimates. In: Proceedings of ACM SIGIR 2003, pp. 111–118 (2003)
3. Brier, G.: Verification of Forecasts Expressed in Terms of Probabilities. Monthly Weather Review 78, 1–3 (1950)
4. DeGroot, M., Fienberg, S.: The Comparison and Evalution of Forecasters. The statistician 32, 12–22 (1983)
5. Fawcett, T., Niculescu-Mizil, A.: PAV and the ROC Convex Hull. Machine Learning 68(1), 97–106 (2007)
6. Ferri, C., Flach, P., Hernandez-Orallo, J.: Improving the AUC of Probabilistic Estimation Trees. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 121–132. Springer, Heidelberg (2003)
7. Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Technical Report HPL-2003-4, HP Labs (2003)
8. Forman, G.: Counting Positives Accurately Despite Inaccurate Classification. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 564–575. Springer, Heidelberg (2005)
9. Greiner, R., Su, X., Shen, B., Zhou, W.: Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. Machine Learning 59(3), 213–235 (2005)
10. Grossman, D., Domingos, P.: Learning Bayesian Network Classifiers by Maximizing Conditional Likelihood. In: Proceedings of ICML 2004, pp. 361–368 (2004)
11. Ling, C.X., Huang, J., Zhang, H.: AUC: A Better Measure than Accuracy in Comparing Learning Algorithms. In: Proceedings of Canadian AI 2003, pp. 329–341 (2003)
12. Margineantu, D.D., Dietterich, T.G.: Improved Class Probability Estimates from Decision Tree Models. Nonlinear Estimation and Classification 171, 169–184 (2002)
13. Provost, F., Domingos, P.: Tree Induction for Probability-Based Ranking. Machine Learning 52, 199–215 (2003)
14. Vanderlooy, S., Hullermeier, E.: A Critical Analysis of Variants of the AUC. Machine Learning 72(3), 247–262 (2008)
15. Wu, S., Flach, P.A., Ferri, C.: An Improved Model Selection Heuristic for AUC. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 478–489. Springer, Heidelberg (2007)
16. Zhang, H., Su, J.: Learning Probability Decision Trees for AUC. Pattern Recognition Letters 27, 892–899 (2006)

A Boosting Approach to Multiview Classification with Cooperation

Sokol Koço and Cécile Capponi*

CNRS – LIF UMR 6166 – Aix-Marseille University
Technopole Chateau-Gombert – 39 rue Joliot Curie,
CMI – 13453 F-Marseille cedex 13
{sokol.koco,cecile.capponi}@lif.univ-mrs.fr
http://www.lif.univ-mrs.fr

Abstract. In many fields, such as bioinformatics or multimedia, data may be described using different sets of features (or views) which carry either global or local information. Some learning tasks make use of these several views in order to improve overall predictive power of classifiers through fusion-based methods. Usually, these approaches rely on a weighted combination of classifiers (or selected descriptions), where classifiers are learned independently. One drawback of these methods is that the classifier learned on one view does not communicate its failures within the other views. This paper deals with a novel approach to integrate multiview information. The proposed algorithm, named Mumbo, is based on boosting. Within the boosting scheme, Mumbo maintains one distribution of examples on each view, and at each round, it learns one weak classifier on each view. Within a view, the distribution of examples evolves both with the ability of the dedicated classifier to deal with examples of the corresponding features space, and with the ability of classifiers in other views to process the same examples within their own description spaces. Hence, the principle is to slightly remove the hard examples from the learning space of one view, while their weights get higher in the other views. This way, we expect that examples are urged to be processed by the most appropriate views, when possible. At the end of the iterative learning process, a final classifier is computed by a weighted combination of selected weak classifiers.

This paper provides the Mumbo algorithm in a multiclass and multiview setting, based on recent theoretical advances in boosting. The boosting properties of Mumbo are proved, as well as some results on its generalization capabilities. Several experimental results are reported which point out that complementary views may actually cooperate under some assumptions.

Keywords: Boosting, Multiview Learning, Supervised Learning, Classification.

* This work is supported by the French agency ANR, project DECODA, contract no 2009-CORD-005-01, and the French business clusters Cap Digital and SCS.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 209–228, 2011.

1 Introduction

In many application domains of machine learning, such as bioinformatics or multimedia indexing, data may be described by several sources or views [1], [2]. When facing a classification or a regression task, the use of these views might be of great interest, since each view is supposed to carry some information that the other views would not embed. Fortunately, there exist many methods to select the most informative sources, or set of features, that either best discriminate data concepts or best describe one concept among others [3] [4].

Most of these selective methods are statistically founded, which means that they tend to disregard localized – isolated – information although it could be useful to compensate the lack of performance on some (group of) learning examples. Indeed, real-life data descriptions are often noisy. When the noise rate of a set of feature descriptions reaches a threshold, which depends on the problem, no learning algorithm has been proved, neither theoretically nor empirically, to be able to overcome the noise disruption on the generalization capabilities of classifiers. Yet, multiview learning approaches should enable some localized failures of classifiers trained on one view, to be compensated by – or subordinated to – the abilities of classifiers on the other views.

Up to now, several approaches of multiview learning have been developed, most of them in the semi-supervised setting. The first of them was the well-known *Co-Training* algorithm [5], which was based on far too much restrictive assumptions [6]. Other semi-supervised multiview algorithms have then been developed and theoretically founded, all of them promoting the agreement between views [7] [8] [9], except for the semi-supervised boosting approach presented in [10]. No compelling application on real-life problems has promoted these approaches yet, although many problems would actually need a multiview learning process.

In addition, in the supervised setting, leveraging the performances of classifiers learned on different views has mainly been performed through fusion-based methods, either early or late fusion [11] [12]. Early fusion consists in grouping (selected) descriptions of the different views into a large vector, and then to learn a classifier on this resulting view. On the opposite, late fusion allows one classifier per view to be learned, while the final classifier is a combination of them. Usually, late fusion performs better than early fusion. Yet, none of them leads to good performances when the views are of unbalanced informative content, for weaker views tend to reduce the final performances. An empirical comparison of these methods applied on multimedia problems is presented in [13].

Whatever the fusion-based approach is, it relies on a weighted combination of classifiers (or selected descriptions), where classifiers are learned independently. One drawback of these methods is that the classifier learned on one view does not communicate its failures to the other views. Besides, views must be independent in order for the combined classifier to be most accurate.

Yet, we think it could be interesting that, when the classifier on a view fails on a region of examples in the instance space, it could entrust the other views with the classification of these examples. One of the major difficulties is then

to delimit the concerned subareas of the instance space, without loss of generalization capabilities. Instead of precisely locating these subareas, we propose an algorithm based on boosting [14] [15] whose principle is to slightly remove the hard examples from the learning space of one view, while their weights get higher in the other views. This way, we expect that examples are processed by the most appropriate views.

In order to implement this principle, we designed Mumbo as a boosting-like algorithm which works from different views on data. Each view is associated with a weak learner, and one distribution per view is maintained (section 2). The distributions are updated at each iteration in such a way that views communicate the ones to the others their capability of processing learning examples. Hence, not only the distribution update in one view takes into account the performances of that view in classifying the learning examples, but it also embeds the performances of the other views on these examples. The properties of Mumbo are discussed and proved in section 3: both empirical and generalization errors are bounded. In order to warrant the boosting properties, and the generalization error bound, we define a global distribution of examples that reflects the overall behaviour of the algorithm within a given hypothesis space. In section 4, we present experimental results of Mumbo on synthetic data, which confirm that Mumbo is a boosting algorithm, better than other basic fusion approaches. Before concluding, we discuss this approach with other methods, and give some clues to improve Mumbo (section 5).

2 The Mumbo Algorithm

2.1 Principles and Assumptions

Mumbo is a multiview boosting algorithm: each example of the learning sample S is represented by several independent sets of features. Each one of these representations is called a *view*. Eventually, these views are used to train models, which are then used to classify other examples. Even though these models are learned on different representations of the same examples, they are by no means equal performance wise. Classifiers learned on some views perform better than those learned on other views, due to the noise in the data and/or views, or the lack of information, etc., which may be different from one representation space to another. In other words, we may define the strength of a view as the possibility to learn a good classifier on that view. At the same time, the weakness of a view may reflect the impossibility of learning a good classifier from the instance space defined by this view.

More formally, let S be a sample of n tagged examples chosen according to some distribution $\mathcal{D}$ over $X \times Y$, where X is some instance space and Y is some class space. Let V be a view, $\mathcal{H}$ be the space of all the hypothesis that we can learn on V and h be the best classifier that we can learn on this space. Finally, let ρ be the error of random guessing over S and $\sigma_V \leq \rho$ be the lower bound of the error of h on S. We define the notion of weak and strong view as follows : V is called a strong view if σ_V is near 0 and V is called a weak view if $\gamma_V = \rho - \sigma_V$ is near 0.

Mumbo has been designed in order to learn a classifier in a multiview setting, where views are supposed to be of different strengths. More specifically, we suppose that among the views, there exists one strong major view V, and several weaker minor views $v_1, \cdots, v_z$.

In our setting, γ_V is supposed to be greater than $\gamma_{v_1}, \cdots, \gamma_{v_z}$.

For example, in speech recognition, three usual views for describing a speech (or dialog) to be classified, are known to be of unequal strength [16]. The major view is the lexical analysis of a speech (syntactic trees, for example); other minor views may be the prosodic information, and syntactic information. Although the major view allows to learn rather good classifiers, researchers in speech recognition still use minor views for learning in order to compensate the failures of the major view in case of noise disruptions [17].

As pointed out in the introduction, the basic principle of Mumbo is to encourage each view v to focus on the examples that are hard to process in other views, and easy to process in v. Hence, it assumes that if one representation space does not embed information on one (set of) examples, part of that information can be provided by other representation spaces.

2.2 Framework and Notations

In this paper, we present Mumbo within the framework defined by [15], where basically γ denotes the edge of a classifier with regards to random. We use the following typings:

- matrices are denoted by bold capital letters like $\mathbf{C}$; element of row i and column j in matrix $\mathbf{C}$ is denoted $C(i,j)$, and $C(i)$ is the row i of $\mathbf{C}$.
- $\mathbf{M} \cdot \mathbf{M'}$ denotes the Frobenius inner product of two matrices.
- the indicator function is denoted by $\mathbb{1}[\cdot]$, and the cartesian product is denoted by $X_1 \times X_2$.

Let $S = \{(x_1, y_1), ..., (x_n, y_n)\}$ be the learning sample, where $x_i \in X$, and $y_i \in Y$ is the class of the example x_i. The set of classes is $Y = \{1, ..., k\}$. The set of features X is made up of different subsets: $X = X_1 \times ... \times X_m$. Each subset represents a view, as in [18]. Then, the representation of example x_i within view m is written $x_{i,m}$[1].

In this paper we use the definition of weak classifier as defined in [15], that is a classifier whose performance on a cost matrix is better than that of some edge-over-random γ baseline.

Definition 1 (edge-over-random baseline and cost matrix, by Mukherjee et al. [15]). *The edge-over-random baseline* $\mathbf{B} \in \mathcal{B}_\gamma^{eor} \subseteq \mathbb{R}^{n \times k}$*, where* $\mathcal{B}_\gamma^{eor}$ *is the space of edge-over-random baselines, is* γ *more likely to predict the correct label rather than an incorrect one on every example* i*:* $\forall l \neq y_i, B(i, y_i) \geq B(i, l) + \gamma$*, with equality holding for some* l*.*

The edge-over-random cost matrix $\boldsymbol{C}$ *puts the least cost on the correct label,* i.e. *the rows of the cost matrix come from the set* $\left\{c \in \mathbb{R}^k : \forall l, c(y_i) \leq c(l)\right\}$.

[1] When possible, we simplify $x_{i,m}$ to x_i in the scope of view m.

Definition 2 (edge-over-random baseline in the Mumbo setting). *The edge-over-random baseline used in this paper is a cost matrix* $\boldsymbol{U}_\gamma$ *defined as follows:* $\boldsymbol{U}_\gamma(i,l) = (1-\gamma)/k + \gamma$ *if* $l = y_i$ *and* $\boldsymbol{U}_\gamma(i,l) = (1-\gamma)/k$ *if* $l \neq y_i$.

The $\mathbf{1}_{h_{t,m}}$ *matrix is the prediction matrix defined as* $\mathbf{1}_{h_{t,m}}(i,l) = 1$ *if* $h_{t,m}(i) = l$ *and* $\mathbf{1}_{h_{t,m}}(i,l) = 0$ *if* $h_{t,m}(i) \neq l$.

Definition 3 (edge-condition, by Mukherjee et al. [15]). *Let* $\mathbf{C} \subseteq \mathbb{R}^{n\times k}$ *and matrix* $\mathbf{B} \in \mathcal{B}_\gamma^{eor}$*, an eor-baseline; we say that a weak classifier* h *satisfies the edge condition if* $\mathbf{C}\cdot\mathbf{1}_h \leq \mathbf{C}\cdot\mathbf{B}$

In the case of binary classification, the ith row of the baseline $\mathbf{U}_\gamma$ is $(\frac{1}{2}(1-\gamma), \frac{1}{2}(1+\gamma))$ if the label of example i is $+1$, and $(\frac{1}{2}(1+\gamma), \frac{1}{2}(1-\gamma))$ if the label of example i is -1. A given classifier h satisfies the edge condition if $\sum_i \mathbf{C}(i,h(i)) \leq \sum_i \left\{(\frac{1}{2}-\frac{\gamma}{2})\mathbf{C}(i,\overline{y}_i) + (\frac{1}{2}+\frac{\gamma}{2})\mathbf{C}(i,y_i)\right\}$ and [15] shows that this condition is equivalent to the usual weak learning condition for binary classification.

2.3 The Core of Mumbo

Mumbo (algorithm 1) is an attempt to promote the collaboration between major and minor views, in order to enhance the performances of classifiers usually learned only on the major view. It is a boosting algorithm theoretically founded on the framework presented in [15].

Y is not limited to $\{-1,+1\}$, since we are in the multiclass setting, based on the theoretical approach of [15], whose one of the main ideas is to replace the weights of the examples with a cost matrix. We use the same idea here: $\mathbf{C}$ is a cost matrix so that $\mathbf{C}(i,l)$ is the cost of assigning the label l to the example i. Since we deal with more than one view, we use one cost matrix $\mathbf{C}_j$ per view j, and a global cost matrix $\mathbf{C}_G$. Thus $m+1$ cost matrices are maintained.

Mumbo runs for T rounds: at each round t, a weak learner is trained on each view v, which returns m weak classifiers $h_{t,m}$. These weak classifiers must satisfy the weak learning condition given in definition 3. For each $h_{t,j}$, we compute a parameter $\alpha_{t,j}$ that measures its importance depending on the edge of the $h_{t,j}$ on the cost matrix $\mathbf{C}_{t,j}$ (*c.f.* algorithm 1).

As stated before, one of the main ideas of Mumbo is to have some sort of collaboration between the different views. This idea is implemented in two different parts of this algorithm: first during the update of the m cost matrices, and second when choosing the classifier h_t selected at round t in the final combination.

The update of each cost matrix depends on all the classifiers. The i^{th} line, corresponding to the example x_i in the matrix of the view j, is updated only if the classifier learned on this view classifies correctly x_i OR if all the $m-1$ other weak classifiers misclassify it. Intuitively this means that a view gives up on the hardest examples and lets the other views handle them. In the scenario of one major and several minor views, this allows the minor views to focus on the hardest examples of the major view.

Algorithm 1. Mumbo: MUltiModal BOosting

Given

- S = $\{(x_1, y_1), ..., (x_n, y_n)\}$ where $x_i \in X_1 \times X_2 \times ... \times X_m$, $y_i \in \{1, ..., k\}$
- m weak learning algorithms WL
- T the number of iterations
- a baseline $\mathbf{B}$ (edge-over-random prior baseline)

Initialize $(\forall i \in \{1, \cdots, n\}, \forall j \in \{1, \cdots, m\}, \forall l \in \{1, \cdots, k\})$:

$f_{0,j}(i, l) = 0$

$\mathbf{C}_{0,G}(i, l) = \mathbf{C}_{0,j}(i, l) = \begin{cases} 1 & \text{if } y_i \neq l \\ -(k-1) & \text{if } y_i = l \end{cases}$ where $\mathbf{C}_{0,G}$ is the global cost matrix

for $t = 1$ **to** T **do**

Train WL using $\mathbf{C}_{t-1,1}, ..., \mathbf{C}_{t-1,m}$

for $j = 1$ **to** m **do**

Get $h_{t,j}$ satisfying the edge condition on $\mathbf{B}$, and compute edge $\delta_{t,j}$ on $\mathbf{C}_{t-1,j}$, and $\alpha_{t,j} = \frac{1}{2} \ln \frac{1+\delta_{t,j}}{1-\delta_{t,j}}$

end for

Update cost matrices (for each view $j = 1 \cdots m$):

$$\mathbf{C}_{t,j}(i, l) = \begin{cases} \exp(f_{t,j}(i, l) - f_{t,j}(i, y_i)) & \text{if } l \neq y_i \\ -\sum\limits_{p=1; p \neq y_i}^{k} \exp(f_{t,j}(i, p) - f_{t,j}(i, y_i)) & \text{if } l = y_i \end{cases}$$

where $f_{t,j}(i, l) = \sum\limits_{z=1}^{t} \mathbb{1}[h_{z,j}(i) = l] \alpha_{z,j} d_{z,j}(i)$

and $d_{z,j}(i) = \begin{cases} 1 & \text{if } h_{z,j}(i) = y_i \text{ or } \nexists q \in \{1, ..., m\}, h_{z,q}(i) = y_i \\ 0 & \text{else} \end{cases}$

Choose $h_t = \underset{h_{t,j}}{\text{argmax}}(\text{edge } h_{t,j} \text{ on } \mathbf{C}_{t,G})$ and $\delta_t = \{edge\ of\ h_t\ on\ \mathbf{C}_{t,G}\}$

Compute $\alpha_t = \frac{1}{2} \ln \frac{1+\delta_t}{1-\delta_t}$

Update $\mathbf{C}_{t,G}$:

$$\mathbf{C}_{t,G}(i, l) = \begin{cases} \exp(f_{t,G}(i, l) - f_{t,G}(i, y_i)) & \text{if } l \neq y_i \\ -\sum\limits_{j \neq y_i}^{k} \exp(f_{t,G}(i, j) - f_{t,G}(i, y_i)) & \text{if } l = y_i \end{cases}$$

where $f_{t,G}(i, l) = \sum\limits_{z=1}^{t} \mathbb{1}[h_{z,m}(i) = l] \alpha_{z,m}$

end for

Output final hypothesis : $H(x) = \underset{l \in 1, ..., k}{\text{argmax}} f_T(x, l)$, where $f_T(i, l) = \sum\limits_{t=1}^{T} \mathbb{1}[h_t(i) = l] \alpha_{t,m}$

In the last part of each round t, Mumbo chooses the classifier h_t among the m that minimizes the error on the global cost matrix. The confidence α_t is computed for h_t, based on its edge on the global cost matrix. Finally, the global cost matrix is updated, in a similar way that in the adaptive case of the OS algorithm [15].

The final hypothesis H is a weighted vote of the T selected weak classifiers h_t, and α_t is the weight assigned to h_t.

3 Properties of Mumbo

In this section, we present two properties of the Mumbo algorithm that together ensure it is a boosting algorithm. We first show that the update rules for the cost matrix of each view, as presented in the previous section, actually reduce the training error on this view. We then prove that the criterion for choosing the unique h_t at each step t, and eventually the update rule, allows Mumbo to be a safe boosting algorithm: the training error decreases with rounds. The most important property, a bound on the generalization error of Mumbo, is proved.

3.1 Bounding the Training Error on Each View

One property of Mumbo is that we can bound the empirical error made by the final classifiers of one view, when views are considered independently. We give a formal proof of this property, then we define a way of computing $\alpha_{t,m}$ in each round t for each view m in order to prove the decreasing of the empirical error of the final combination of weak classifiers.

Theorem 1 is an adaptation to Mumbo of the lemma presented in the supplement of [15].

Theorem 1 (bounding the empirical error in view m). *For a given view m, suppose the cost matrix $\boldsymbol{C}_{t,m}$ is chosen as in the algorithm 1, and the returned classifier $h_{t,m}$ satisfies the edge condition for the baseline $\boldsymbol{U}_{\gamma_m}$ and cost matrix $\boldsymbol{C}_{t,m}$,* i.e. $\boldsymbol{C}_{t,m} \cdot \boldsymbol{1}_{h_{t,m}} \leq \boldsymbol{C}_{t,m} \cdot \boldsymbol{U}_{\gamma_m}$.

Then choosing a weight $\alpha_{t,m} > 0$ for $h_{t,m}$ makes the error $\epsilon_{t,m} = \sum_{i=1}^{n} \sum_{l \neq y_i} \exp(f_{t,m}(i,l) - f_{t,m}(i,y_i))$, at most a factor

$$\tau_{t,m} = 1 - \frac{1}{2}\left(\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})\right)\delta_{t,m} + \frac{1}{2}\left(\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2\right)$$

of the loss before choosing ($\alpha_{t,m}$), where $\delta_{t,m}$ is the edge of $h_{t,m}$, $\delta_{t,m} = \boldsymbol{C}_{t,m} \cdot \boldsymbol{U}_{\gamma_m} - \boldsymbol{C}_{t,m} \cdot \boldsymbol{1}_{h_{t,m}}$.

Proof

Let S_+ be the set of the examples correctly classified by $h_{t,m}$, S_- the set of the examples misclassified by all the m classifiers returned by WL, and S_{-+} the set of the examples misclassified by $h_{t,m}$ and correctly classified by at least one of the other $h_{t,j}, j \neq m$.

In order to simplify the reading of the proof, we introduce the quantities: $L_{t,m}(i) = \sum_{l \neq y_i} \exp(f_{t,m}(i,l) - f_{t,m}(i,y_i))$, and $\zeta_{t,m}(i,l) = f_{t,m}(i,l) - f_{t,m}(i,y_i)$.

Using the edge condition we have :

$$\mathbf{C}_{t,m} \cdot \mathbf{1}_{h_{t,m}} \leq \mathbf{C}_{t,m} \cdot \mathbf{U}_{\delta_{t,m}} \tag{1}$$

The left and right sides of equation 1 can be rewritten as:

$$\mathbf{C}_{t,m} \cdot \mathbf{1}_{h_{t,m}} = -\sum_{i \in S_+} L_{t-1,m}(i) + \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) + \sum_{i \in S_{-+}} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i)))$$

$$\mathbf{C}_{t,m} \cdot \mathbf{U}_{\delta_{t,m}} = \sum_{i=1}^{N} \left(-L_{t-1,m}(i)(\tfrac{1-\delta_{t,m}}{k} + \delta_{t,m}) \; + L_{t-1,m}(i)(\tfrac{1-\delta_{t,m}}{k}) \right) = -\delta_{t,m} \sum_i L_{t-1,m}(i)$$

So, using the edge condition 1 we obtain:

$$-\sum_{i \in S_+} L_{t-1,m}(i) + \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) + \sum_{i \in S_{-+}} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) \leq -\delta_{t,m} \sum_{i \in S} L_{t-1,m}(i)$$

hence:

$$\sum_{i \in S_+} L_{t-1,m}(i) - \sum_{i \in S_- \cup S_{-+}} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) \geq \delta_{t,m} \sum_{i \in S} L_{t-1,m}(i) \tag{2}$$

In order to compute the drop in loss after choosing $h_{t,m}$ with weight $\alpha_{t,m}$, let us consider three cases:

1. For $i \in S_+$:
   We have $f_{t,m}(i,l) - f_{t,m}(i,y_i) = f_{t-1,m}(i,l) - (f_{t-1,m}(i,y_i) + \alpha_{t,m})$, then:

$$\begin{aligned} \Delta_+ &= \sum_{i \in S_+} -L_{t,m}(i) - \sum_{i \in S_+} -L_{t-1,m}(i) = \sum_{i \in S_+} -\exp(-\alpha_{t,m}) L_{t-1,m}(i) - \sum_{i \in S_+} -L_{t-1,m}(i) \\ &= (1 - \exp(-\alpha_{t,m})) \sum_{i \in S_+} L_{t-1,m}(i) \end{aligned}$$

2. For $i \in S_-$:

$$\begin{aligned} \Delta_- &= \sum_{i \in S_-} \exp(f_{t,m}(i, h_{t,m}(i)) - f_{t,m}(i,y_i)) - \sum_{i \in S_-} \exp(f_{t,m}(i-1, h_{t,m}(i)) - f_{t-1,m}(i,y_i)) \\ &= \sum_{i \in S_-} \exp(f_{t-1,m}(i, h_{t,m}(i)) + \alpha_{t,m} - f_{t-1,m}(i,y_i)) - \sum_{i \in S_-} \exp(f_{t-1,m}(i, h_{t,m}(i)) - f_{t-1,m}(i,y_i)) \\ &= (\exp(\alpha_{t,m}) - 1) \sum_{i \in S_-} \exp(f_{t-1,m}(i, h_{t,m}(i)) - f_{t-1,m}(i,y_i)) \\ &= (\exp(\alpha_{t,m}) - 1) \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) \end{aligned}$$

3. For $i \in S_{-+}$:

$$\begin{aligned} \Delta_{-+} &= \sum_{i \in S_-} \exp(f_{t,m}(i, h_{t,m}(i)) - f_{t,m}(i,y_i)) - \sum_{i \in S_-} \exp(f_{t-1,m}(i, h_{t,m}(i)) - f_{t-1,m}(i,y_i)) \\ &= \sum_{i \in S_-} \exp(\zeta_{t,m}(i, h_{t,m}(x_i))) - \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(x_i))) \\ &= 0 \text{ since the value of } f_{t,m} \text{ does not change for these examples} \end{aligned}$$

So, the drop in loss $\Delta = \Delta_+ - \Delta_- - \Delta_{-+}$ is:

$$
\begin{aligned}
&= (1 - \exp(-\alpha_{t,m})) \sum_{i \in S_+} L_{t-1,m}(i) - (\exp(\alpha_{t,m}) - 1) \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \\
&= \left(\tfrac{\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})}{2}\right) \left(\sum_{i \in S_+} L_{t-1,m}(i) - \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \right) \\
&\quad - \left(\tfrac{\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2}{2}\right) \left(\sum_{i \in S_+} L_{t-1,m}(i) + \sum_{i \in S_-} \exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \right) \\
&\geq \left(\tfrac{\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})}{2}\right) \left(\sum_{i \in S_+} L_{t-1,m}(i) - \sum_{i \in S_- \cup S_{-+}} \exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \right) \\
&\quad - \left(\tfrac{\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2}{2}\right) \left(\sum_{i \in S_+} L_{t-1,m}(i) + \sum_{i \in S_- \cup S_{-+}} \exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \right)
\end{aligned}
$$

Using the result we obtained in equation 2 and the fact that $\exp(\zeta_{t-1,m}(i, h_{t,m}(i))) \leq L_{t-1,m}(i)$, we can give a lower bound of the loss drop:

$$
\begin{aligned}
\Delta &\geq \left(\tfrac{\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})}{2}\right) \delta_{t,m} \sum_i L_{t-1,m}(i) \\
&\quad - \left(\tfrac{\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2}{2}\right) \left(\sum_{i \in S_+} L_{t-1,m}(i) + \sum_{i \in S_- \cup S_{-+}} L_{t-1,m}(i) \right) \\
&\geq \left(\tfrac{\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})}{2}\right) \delta_{t,m} \sum_i L_{t-1,m}(i) - \left(\tfrac{\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2}{2}\right) \sum_i L_{t-1,m}(i) \\
&\geq \left(\tfrac{\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m})}{2} \delta_{t,m} - \tfrac{\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2}{2}\right) \sum_i L_{t-1,m}(i)
\end{aligned}
$$

Hence the loss $1 - \Delta$ at round t is at most a factor $1 - \frac{1}{2}(\exp(\alpha_{t,m}) - \exp(-\alpha_{t,m}))\delta_{t,m} + \frac{1}{2}(\exp(\alpha_{t,m}) + \exp(-\alpha_{t,m}) - 2)$ of the loss in round $t - 1$. □

We proved that, in each view, the training error (cost) decreases. Based on theorem 1, and tuning $\alpha_{t,m}$ to $\frac{1}{2} \ln \frac{1+\delta_{t,m}}{1-\delta_{t,m}}$, we get the following bound on the empirical error of the classifier H_m obtained by the weighted combination of weak classifiers learned in view m after T iterations:

$$
\epsilon_{T,m} \leq (k-1) \prod_{t=1}^{T} \sqrt{1 - \delta_{t,m}} \leq (k-1) \exp\left\{ -\frac{1}{2} \sum_{t=1}^{T} \delta_{t,m}^2 \right\} \tag{3}
$$

This result shows that Mumbo is a boosting algorithm even when the selected weak classifier always comes from the same view m for all steps. This might occur when the major view is far better than minor views for all training examples.

3.2 Bounding the Whole Empirical Error

At each step t of the algorithm 1, one classifier is selected among m weak classifiers, if m is the number of views, that is, the space of weak hypothesis $\mathcal{H}$ in this case is $\{h_{t,1}, \cdots, h_{t,m}\}$. This space is a particular case of the space of hypothesis

used by the OS algorithm, thus we obtain the same bound on the empirical error as the OS algorithm, that is :

$$\epsilon_T \leq (k-1)\prod_{t=1}^{T}\sqrt{1-\delta_{t,m}} \leq (k-1)\exp\left\{-\frac{1}{2}\sum_{t=1}^{T}\delta_{t,m}^2\right\} \tag{4}$$

In practice, one may observe that the edges of the classifiers at step t are all negative. In such a case, since each weak classifier of view v is trained on a subset of the learning samples randomly drawn from the current distribution of v, iterating the learning step until γ_v is positive allows the algorithm to fulfill the conditions.

3.3 Results in Generalization

We show here that the generalization error of the final hypothesis learned by Mumbo after T iterations can be bound, and this bound converges towards 0 with the number of iterations.

The generalization error of a classifier is defined as the probability to misclassify any new example. For multiclass algorithms such as AdaBoost.MR, [19] shows that the generalization error of the final hypothesis can be bound and that it is related to the margins of the learning examples. We thus first recall the definitions of the bound on the generalization error, then we extend existing results to Mumbo.

Generalization Error for Multiclass Problems. The final hypothesis of Mumbo is a multi class classifier, thus its output space can be defined as $Y = \{1, 2, ..., k\}$. In this section, the weak classifiers $h \in \mathcal{H}$ are defined as mappings from $X \times Y$ to $\{0, 1\}$, where X is some description space. The label y is predicted as a potential label for x_i if $h(x, y) = 1$. Note that these classifiers are equivalent to $\mathbb{1}[h_t(x) = l]$, the weak classifiers described in the algorithm.

Let $\mathcal{C}$ denote the convex hull of $\mathcal{H}$, that is :

$$\mathcal{C} = \left\{ f : (x, y) \rightarrow \sum_{h \in \mathcal{H}} \alpha_h h(x, y) | \alpha_h \geq 0 \; and \sum_h \alpha_h = 1 \right\}$$

For a given example x and a label y, a classifier f in $\mathcal{C}$ predicts y as the class of x if $\underset{l \in Y}{argmax} f(x, l) = y$. The margin of an example is then defined as :

$$margin(f, x, y) = f(x, y) - \max_{l \neq y} f(x, l)$$

The function f misclassifies an example x if the margin given by f on the couple (x, y) is negative or zero.

Using the previous definitions, Schapire et al. give a proof of theorem 2 [19]:

Theorem 2 (Schapire *et al.*, [19]). *Let $\mathcal{D}$ be a distribution over $X \times Y$, and let S be a sample of n examples chosen independently at random according to*

$\mathcal{D}$. Assume that the base-classifier[2] space $\mathcal{H}$ is finite, and let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S, every function $f \in \mathcal{C}$ satisfies the following bound for all $\theta > 0$:

$$\mathbf{P}_{\mathcal{D}}[margin(f,x,y) \leq 0] \leq \mathbf{P}_S[margin(f,x,y) \leq \theta] + O\left(\frac{1}{\sqrt{n}} + \left(\frac{\log(nk)\log(|\mathcal{H}|)}{\theta^2} + log(1/\delta)\right)^{1/2}\right)$$

More generally, for finite or infinite $\mathcal{H}$ with VC-dimension d, the following bound holds as well, assuming that $n \leq d \leq 1$:

$$\mathbf{P}_{\mathcal{D}}[margin(f,x,y) \leq 0] \leq \mathbf{P}_S[margin(f,x,y) \leq \theta] + O\left(\frac{1}{\sqrt{n}} + \left(\frac{d\ log^2(nk/d)}{\theta^2} + log(1/\delta)\right)^{1/2}\right)$$

In theorem 2, the term $\mathbf{P}_{\mathcal{D}}[margin(f,x,y) \leq 0]$ is the generalization error of the function f. The term $\mathbf{P}_S[margin(f,x,y) \leq \theta]$ is the empirical margin error of f on the sample S, that is, the proportion of examples of S which are misclassified, or which are correctly classified but with a margin smaller than θ. In the following section, we use $\epsilon_\theta(f,S)$ instead of $\mathbf{P}_S[margin(f,x,y) \leq \theta]$.

The second term in the theorem is a complexity penalization cost.

Mumbo. Theorem 2 holds for every voting method using multiclass classifiers as weak classifiers; it thus also holds for Mumbo since his final hypothesis is $H_T(x) = \underset{l \in 1,2,...,k}{argmax}\ f_T(x,l)$, where :

$$f_T(x,l) = \left(\sum_{t=1}^{T} h_t(x,l)\alpha_t\right) / \sum_{t=1}^{T}\alpha_t$$

The weak classifier h_t chosen at each iteration is selected from a set of classifiers $\{h_{t,1},...,h_{t,m}\}$. These classifiers are selected from potentially different spaces of hypothesis, namely $\mathcal{H}_1,...,\mathcal{H}_m$. Thus the space of hypothesis $\mathcal{H}$ from which h_t is selected is the union of $\mathcal{H}_1,...,\mathcal{H}_m$. We deduce by the definition of the VC-dimension [20] that $d_{\mathcal{H}} = min\{d_{\mathcal{H}_1},...,d_{\mathcal{H}_m}\}$.

We still have to prove that the generalization error decreases with the number of iterations. To do so, it is sufficient to prove that the empirical margin error decreases, since the term $O\left(\frac{1}{\sqrt{n}} + \left(\frac{d\ log^2(nk/d)}{\theta^2} + log(1/\delta)\right)^{1/2}\right)$ is a constant.

We start with showing that we can find a bound for $\epsilon_\theta(f_T,S)$.

Lemma 1. *The empirical margin error of Mumbo after T iterations is bounded by:*

$$\epsilon_\theta(f_T,S) \leq \frac{(k-1)}{n}\left(\prod_{t=1}^{T}(1+\delta_t)^{\frac{1+\theta}{2}}(1-\delta_t)^{\frac{1-\theta}{2}}\right)$$

[2] Note : the base-classifiers are referred to as weak classifiers in our paper.

Proof

Let $l = \underset{y' \neq y}{argmax} f(x, y')$. For readability reasons, we may write $\sum_t$ instead of $\sum_{t=1}^{T}$.

By the definition of the margin and f, we get :

$$margin(f, x, y) = f(x, y) - f(x, l) = \frac{\sum_t h_t(x, y)\alpha_t}{\sum_t \alpha_t} - \frac{\sum_t h_t(x, l)\alpha_t}{\sum_t \alpha_t}$$

Hence,

$$\begin{aligned} margin(f, x, y) < \theta &\Leftrightarrow \frac{\sum_t h_t(x,y)\alpha_t}{\sum_t \alpha_t} - \frac{\sum_t h_t(x,l)\alpha_t}{\sum_t \alpha_t} \leq \theta \\ &\Leftrightarrow \theta \sum_t \alpha_t - \left(\sum_t h_t(x, y)\alpha_t - \sum_t h_t(x, l)\alpha_t \right) \geq 0 \end{aligned}$$

Let $A_i = -\left(\sum_t \alpha_t h_t(x_i, y) - \sum_t \alpha_t h_t(x_i, l) \right)$ and $B = \theta \sum_t \alpha_t$. We deduce that $\mathbf{P}[margin(f, x_i, y) \leq \theta] = 1 \Leftrightarrow A_i + B \geq 0$, that is, $\exp(A_i + B) \geq \mathbf{P}[margin(f, x, y) \leq \theta]$. Thus, $\epsilon_\theta(f_T, S) \leq \frac{1}{n} \sum_{i=1}^{n} \exp(A_i) \exp(B)$.

$$\begin{aligned} \epsilon_\theta(f_T, S) &\leq \tfrac{1}{n} \sum_{i=1}^{n} \exp(A_i) \exp(B) \\ &\leq \tfrac{1}{n} \sum_{i=1}^{n} \exp\left(-(\sum_t \alpha_t h_t(x_i, y) - \sum_t \alpha_t h_t(x_i, l)) \right) \exp(\theta \sum_t \alpha_t) \\ &\leq \tfrac{1}{n} \sum_{i=1}^{n} \exp\left(-(f_T(x_i, y) - f_T(x_i, l)) \right) \exp(\theta \sum_t \alpha_t) \\ &\leq \tfrac{1}{n} \sum_{i=1}^{n} \sum_{y' \neq y} \exp\left(f_T(x_i, y') - f_T(x_i, y) \right) \exp(\theta \sum_t \alpha_t) \end{aligned}$$

Using the bound on the empirical error, we deduce :

$$\begin{aligned} \epsilon_\theta(f_T, S) &\leq \tfrac{1}{n} \exp(\theta \sum_t \alpha_t)(k-1) \prod_t \sqrt{1-\delta_t^2} \leq \tfrac{1}{n} \exp(\theta \sum_t \tfrac{1}{2} \ln(\tfrac{1+\delta_t}{1-\delta_t}))(k-1) \prod_t \sqrt{1-\delta_t^2} \\ &\leq \tfrac{1}{n} \prod_t (\tfrac{1+\delta_t}{1-\delta_t})^{\frac{\theta}{2}} (k-1) \prod_t \sqrt{1-\delta_t^2} \leq \tfrac{(k-1)}{n} \left(\prod_t (1+\delta_t)^{\frac{1+\theta}{2}} (1-\delta_t)^{\frac{1-\theta}{2}} \right) \end{aligned}$$ □

The lemma 1 gives a bound on the empirical margin error. As it was shown in [19], if $\theta < \delta_t/2$, then $(1+\delta_t)^{\frac{1+\theta}{2}}(1-\delta_t)^{\frac{1-\theta}{2}} < 1$. We thus finally claim that the generalization error decreases with the number of iterations:

Theorem 3. *Let $\theta > 0$ be a fixed margin, then the empirical margin error $\epsilon_\theta(f_T, S)$ converges towards 0 with the number of iterations, if the edge of the weak hypothesis selected at each iteration is $> 2\theta$.*

Theorem 3 and the bound given in theorem 2 together prove that *the generalization error of the final hypothesis of Mumbo* decreases with the number of iterations. Indeed, the second term of the bound in theorem 2 is a constant, since all the parameters, including $d_{\mathcal{H}}$, are constant in a given problem, and theorem 3 proves that the first term of the bound decreases with the number of iterations.

4 Experiments on Mumbo

In order to empirically validate and illustrate this approach of multiview learning with boosting, we mainly used synthetic data that obey the underlying assumptions of Mumbo. After explaining the used protocol, this section presents and discusses the results of experiments.

4.1 Protocols

Data Generation. Data is generated within 3 views, and clustered in two classes $\{-1, +1\}$. In each view, the descriptions of examples are vectors of real values. Examples of each class y in view v are generated along a gaussian distribution $\mathcal{G}[m_{y,v}, \sigma_{y,v}]$. However, in order to generate weak views, two types of noise disrupt the sample:

- in each view, the distributions of classes may overlap: some examples are likely to belong to both classes[3].
- In each view, some examples are generated using a uniform distribution, the same for both classes. Let η be the rate of such a description noise (η_M is the noise rate of the major view, while η_m is the noise rate of minor views).

One major view is generated. The two minor views are generated with $\eta_m = \frac{3-2\eta_M}{4}$ in such a way that half of the noisy examples in view M are likely to be sane in minor views. Figure 1 pictures an example of a learning sample with $n = |S| = 20$ examples per class.

We can associate the disruption amount (distribution overlap and noise on descriptions) with the edge-over-random capabilities of weak-classifiers. The more disruption we have in a view, the more γ_v is low on that view. Such a sample generation process was designed in order to fit the assumptions that lead to the design of Mumbo: views are rather weak, and learning a classifier on the whole sample needs a cooperation between learners on each view, because information may be distributed among views.

Processing Experiments. Each weak classifier on view v is obtained by training a linear SVM on a subsample of examples randomly drawn from the current distribution (cost matrix) of v. We check that each weak classifier trained on the view v complies with the definition of weak classifiers in the theoretical scheme of [15], using **B**=**U**. Results are the mean of 10 experiments: one experiment

[3] For these examples x, $P(y = +1|x) = P(y = -1|x)$).

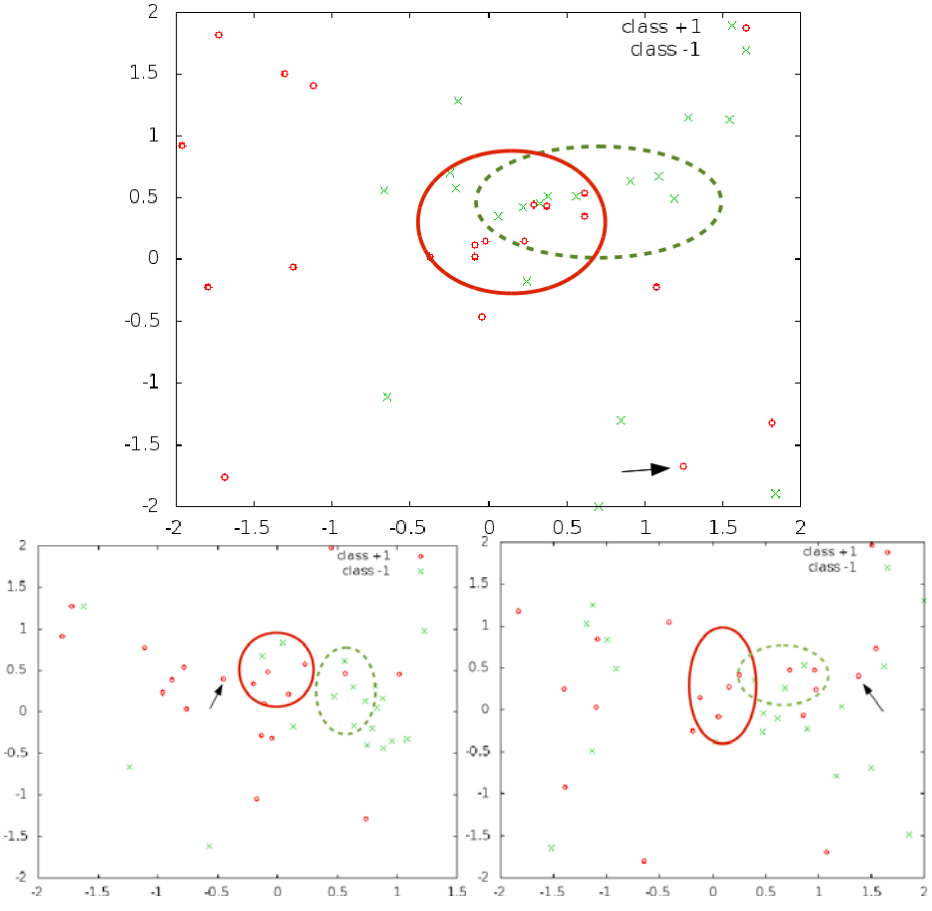


Fig. 1. Each example of the learning sample is represented under three views: the major view is on top, with $\eta_M = 0.38$; other views are minor (bottom), with $\eta_m = 0.56$. The ovals picture the parameters of the examples distribution within each class. The same example is pointed out in each view, in order to illustrate the distribution of information among views.

is made up of (1) the generation of learning and test samples, (2) the learning process, and (3) the evaluation process.

As said in the introduction, Mumbo was designed as an alternative way to fuse classifiers. We thus compare it with two basic methods of fusion, and with Adaboost: (1) late fusion SVM: one RBF SVM is trained on each view, and the final decision is a margin-weighted combination of their results; (2) early fusion SVM: descriptions of each example are concatenated, then a RBF SVM is trained on the single resulting view; and (3) early fusion Adaboost: descriptions of each example are concatenated, then Adaboost is trained on the single resulting view, with a RBF SVM on a subsample of examples as the weak learner.

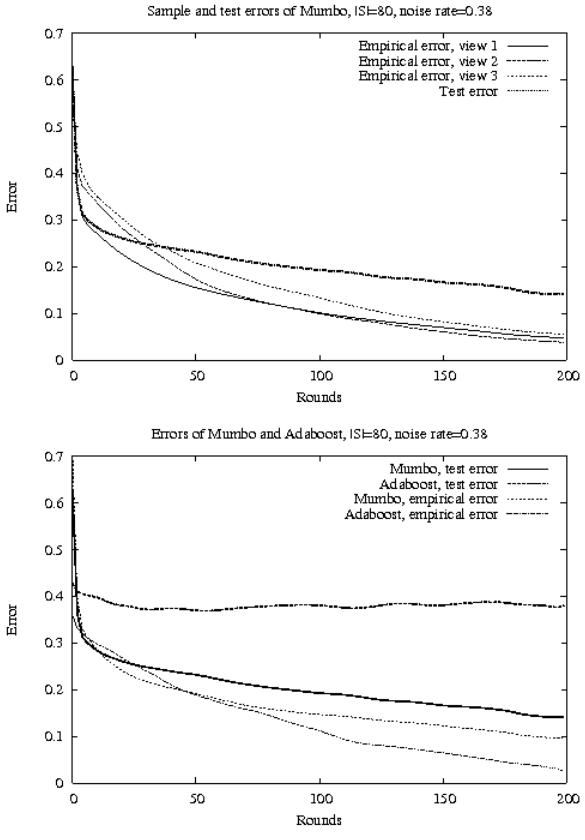


Fig. 2. Empirical and test errors of Mumbo (top), and Mumbo vs. Adaboost early fusion

Classifiers performances are computed using a testing sample drawn from the same setting that generated the learning sample, but twice bigger.

4.2 Results

We present here two kinds of results: an illustration of the behaviour of Mumbo, and a comparison of Mumbo with basic fusion approaches.

Illustration of Boosting Properties. Figure 2 reports, on the left, the boosting-like behaviour of Mumbo. As expected, the empirical costs on each view decrease with iterations, and the estimation of the generalization error also decreases. On the right, the figure pictures a first comparison of Mumbo with Adaboost (in an early fusion setting). We obtained this results with $n = |S| = 60$ and $\eta_M = 0.12$, but the same outlines of behaviours are observed whatever the parameters are ($|S|$ from 20 to 200, and η_M from 0 to 0.5).

Table 1. Comparison of Mumbo with early fusion and late fusion (base classifiers RBF SVM). Note that results of Adaboost are given after 200 iterations (like Mumbo): raw results show that Adaboost obtains slightly better results after about 50 iterations, then tends to over-fit.

$\lvert S\rvert = 80$, η_M	0.5	0.38	0.25	0.12	0
Early+SVM	0.390	0.410	0.437	0.396	0.389
SVM+Late	0.246	0.229	0.263	0.254	0.232
Early+Adaboost	0.415	0.420	0.403	0.364	0.358
Mumbo	0.148	0.152	0.168	0.174	0.164

$\lvert S\rvert = 120$, η_M	0.5	0.38	0.25	0.12	0
Early+SVM	0.367	0.382	0.396	0.389	0.343
SVM+Late	0.198	0.225	0.240	0.208	0.279
Early+Adaboost	0.425	0.415	0.466	0.411	0.389
Mumbo	0.02	0.036	0.012	0.026	0.020

The bad results of Adaboost are not surprising. Indeed, it processes examples on only one view that concatenates the smaller views. Since data was generated such that half of the disrupted examples on the major view are not disrupted in the minor views, the concatenation of descriptions leads to about 75% of noisy data. Adaboost is well-known to be sensitive to the noise, so one cannot expect better results, despite the true convergence of its empirical error.

In addition, which is not reported here, we observed that, whatever η_M is (always under 0.5), weak classifiers on minor views are selected in some rounds, in addition to the weak classifiers of the major view which are the most often selected. First rounds tend to only select the classifiers of the major view, then the minor views are alternatively selected with the major view. Besides, this behaviour can be observed on the first rounds on figure 2. It empirically shows that Mumbo actually encourages views to cooperate.

Comparison with Other Approaches. Table 1 compares Mumbo with basic early and late fusion approaches, with various values of η_M and various sizes of S. Late SVM is the best fusion approach with this type of data, which is not surprising since data is partially noisy (either in description or because distributions overlaps). Yet Mumbo is better for it processes the cooperation among views, leading each view to focus on the examples disrupted in other views.

However, the learning time of Mumbo is T times longer than the learning time of Late SVM. The collaboration slightly improves the results when the major view is disrupted. This is quite obvious with smaller learning samples (when $|S|$=15 or 30).

4.3 Discussion

As expected theoretically, the boosting usual behaviour is observed throughout the experiments, and the results of Mumbo are very good on synthetic data. These results validate the relevance of the Mumbo algorithm when cooperation among views is mandatory for obtaining a strong classifier. Results on empirical and generalization bounds of section 3 are also observed.

In further works, we should test Mumbo on UCI benchmarks. However, these benchmarks are not designed for multiview learning. We plan to select

relevant views on these benchmarks (one major and several minor) using PCA or Canonical Component Analysis tools.

5 Related Works and Discussion

5.1 Related Works

So far, in the supervised setting, there is no multiview machine learning algorithm that considers the representation spaces as complementary. Early and late fusion-based approaches are only empirical ways to process the whole useful information available on samples.

The Multiple Kernel Learning (MKL) approaches [21], which may be used to process multiview samples, is then a costing way to rank the views. But yet, MKL does not promote the cooperation between views: it is much like a way to select the strongest views.

The closest approaches to Mumbo are co-training [5] and 2-BOOST [18]. The former is a multiview approach in the *semi-supervised setting*, where views iteratively cooperate for producing classifiers that converge to the same final hypothesis. The latter is a multiview boosting algorithm. However, Mumbo is different from co-training, first because it works in the supervised setting, and second because it does not assume that the classifiers actually must agree on the same examples. Indeed, Mumbo exploits the disagreements between views. Mumbo is thus closer to 2-BOOST, although the motivations are not the same. 2-BOOST is designed for dealing with one specific weak learner per view, in order to manage homogeneous views. Then, 2-BOOST maintains only one global distribution of examples, whereas Mumbo maintains as many distributions as views in order to process cooperation.

Mumbo is an algorithm that may be categorized as an *ensemble of classifiers*, for the final classifier is a combination of other classifiers. In the literature, it was proved that without diversity between combined classifiers, the resulting classifier can not be better than the best of the combined classifiers. Many measures of diversity have then been studied so far [22]. We think that the hypothesis underlying Mumbo promote such a diversity. In that sense, we aim at obtaining some theoretical results between some diversity measures and classification accuracy of Mumbo.

5.2 Discussion and Improvements

In algorithm 1, the function $d_z(\cdot)$ indicates whether the update of the cost matrix is possible or not. It is a discrete $0-1$ function, which allows the update of the cost matrix only when some conditions are met (section 2.3). However, such an update rule might be too drastic for promoting the collaboration between views. We think that smoothing it, by changing its range to $[0, 1]$, could improve the efficiency of the cooperation. Hence, in addition of having one or several views, each specialized in some parts of the description space, the minor views could be used more efficiently to increase the accuracy of the final predictions.

One of the main ideas of Mumbo is to enhance as much collaboration as possible between the views. We believe that it is possible to achieve a better cooperation also by changing the decision rule for h_t. Indeed, in algorithm 1, h_t is the best classifier among the m chosen classifiers at round t, *i.e.* the one that guarantees the best edge on the general cost matrix $G_{t,m}$. Many other ways to choose h_t could be studied, namely a combination of a subset of weak classifiers, or the choice between the weak classifier of the major view and a combination of the classifiers on minor views, etc. Hence, many alternate selections deserve to be studied, both theoretically (for example, in the PAC-Bayes framework [23]) and empirically.

The most urging work on Mumbo is its study on benchmark and real data. In some domains, such as image indexing, the views are quite natural: there exists dozens of image descriptors, either global or local, that could be considered as complementary views (texture vs. color, etc.). In many other domains, though, we must select the views according to the hypothesis underlying Mumbo (one major still weak view, and many minor views). We wish to adapt statistical tools for view selection, such as Principal Component Analysis as the simplest one.

6 Conclusion and Future Works

Mumbo is a boosting-like algorithm in the setting of multiview learning, where views are of different strenghts with regard to a classification task. The idea underlying Mumbo is to promote the cooperation between stronger and weaker views. To implement this idea, the originality of Mumbo is to maintain one distribution of examples per view, and to proceed to distribution updates that allow some views to focus on examples that are hard to classify in other views.

Mumbo is proved to be a boosting algorithm, within the new theoretical framework of [15]: the empirical error decreases with iterations, globally and within each view. Then, the generalization error of Mumbo is proved to be bounded. Finally, the experimental results on dedicated synthetic data give credits to the relevance of Mumbo for encouraging the cooperation among complementary views.

For now, Mumbo is a first attempt to tackle the problem of unbalanced views on data, and we expect to improve it both theoretically and through experiments on benchmarks and real data.

References

1. Masulli, F., Mitra, S.: Natural computing methods in bioinformatics: A survey. Information Fusion 10(3), 211–216 (2009)
2. Mansoorizadeh, M., Charkari, N.M.: Multimodal information fusion application to human emotion recognition from face and speech. Multimedia Tools and Applications 49(2), 277–297 (2010)
3. Culp, M., Michailidis, G., Johnson, K.: On multi-view learning with additive models. Annals of Applied Statistics 3, 292–318 (2009)

4. Sridharan, K., Kakade, S.M.: An information theoretic framework for multi-view learning. In: Annual Conference on Computational Learning Theory, pp. 403–414 (2008)
5. Blum, A.B., Mitchell, T.M.: Combining labeled and unlabeled data with co-training. In: 11th Annual Conference on Computational Learning Theory, pp. 92–100 (1998)
6. Christoudias, M.C., Urtasun, R., Kapoor, A., Darrell, T.: Co-training with noisy perceptual observations. In: Computer Vision and Pattern Recognition, pp. 2844–2851 (2009)
7. Sindhwani, V., Rosenberg, D.S.: An rkhs for multi-view learning and manifold co-regularization. In: International Conference on Machine Learning, pp. 976–983 (2008)
8. Christoudias, M.C., Urtasun, R., Darrell, T.: Multi-view learning in the presence of view disagreement. In: 24th Conference on Uncertainty in Artificial Intelligence, pp. 88–96 (2008)
9. Ambrym-Maillard, O., Vayatis, N.: Complexity versus agreement for many views: Co-regularization for multi-view semi-supervised learning. In: 20th International Conference on Algorithmic Learning Theory, pp. 232–246 (2009)
10. Saffari, A., Leistner, C., Godec, M., Bischof, H.: Robust multi-view boosting with priors. In: Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III, ECCV 2010, Heraklion, Crete, Greece, pp. 776–789. Springer-Verlag, Berlin, Heidelberg (2010)
11. Kludas, J., Bruno, E., Marchand-Maillet, S.: Information fusion in multimedia information retrieval. In: Boujemaa, N., Detyniecki, M., Nürnberger, A. (eds.) AMR 2007. LNCS, vol. 4918, pp. 147–159. Springer, Heidelberg (2008)
12. Wozniak, M., Jackowski, K.: Some remarks on chosen methods of classifier fusion based on weighted voting. In: Corchado, E., Wu, X., Oja, E., Herrero, Á., Baruque, B. (eds.) HAIS 2009. LNCS, vol. 5572, pp. 541–548. Springer, Heidelberg (2009)
13. Snoek, C., Worring, M., Smeulders, A.: Early versus late fusion in semantic video analysis. In: Proceedings of the 13th Annual ACM International Conference on Multimedia, MULTIMEDIA 2005, pp. 399–402. ACM, New York (2005)
14. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning, pp. 148–156 (1996)
15. Mukherjee, I., Schapire, R.E.: A theory of multiclass boosting. In: Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R.S., Culotta, A. (eds.) Advances in Neural Information Processing Systems, vol. 23, pp. 1714–1722 (2010)
16. Goldwater, S., Jurafsky, D., Manning, C.D.: Which words are hard to recognize? prosodic, lexical, and disfluency factors that increase speech recognition error rates. Speech Communication 52, 181–200 (2010)
17. Hasegawa-Johnson, M., Chen, K., Cole, J., Borys, S., Kim, S.-S., Cohen, A., Zhang, T., Choi, J.-Y., Kim, H., Yoon, T., Chavara, S.: Simultaneous recognition of words and prosody in the boston university radio speech corpus. Speech Communication 46, 418–439 (2005)
18. Janodet, J.-C., Sebban, M., Suchier, H.-M.: Boosting classifiers built from different subsets of features. Fundam. Inf. 96, 89–109 (2009)
19. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The annals of statistics 26(5), 1651–1686 (1998)
20. Vapnik, V.N.: Statistical Learning Theory. Wiley-Interscience, Hoboken (1998)

21. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., El Ghaoui, L., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. J. Mach. Learn. Res. 5, 27–72 (2004)
22. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Machine Learning 51(2), 181–207 (2003)
23. Germain, P., Lacasse, A., Laviolette, F., Marchand, M.: Pac-bayesian learning of linear classifiers. In: Proceedings of the 26th Annual International Conference of Machine Learning, pp. 353–360 (2009)

ARTEMIS: Assessing the Similarity of Event-Interval Sequences

Orestis Kostakis, Panagiotis Papapetrou, and Jaakko Hollmén

Department of Information and Computer Science, Aalto University, Finland,
Helsinki Institute for Information Technology, Finland

Abstract. In several application domains, such as sign language, medicine, and sensor networks, events are not necessarily instantaneous but they can have a time duration. Sequences of interval-based events may contain useful domain knowledge; thus, searching, indexing, and mining such sequences is crucial. We introduce two distance measures for comparing sequences of interval-based events which can be used for several data mining tasks such as classification and clustering. The first measure maps each sequence of interval-based events to a set of vectors that hold information about all concurrent events. These sets are then compared using an existing dynamic programming method. The second method, called `Artemis`, finds correspondence between intervals by mapping the two sequences into a bipartite graph. Similarity is inferred by employing the Hungarian algorithm. In addition, we present a linear-time lower-bound for `Artemis`. The performance of both measures is tested on data from three domains: sign language, medicine, and sensor networks. Experiments show the superiority of `Artemis` in terms of robustness to high levels of artificially introduced noise.

Keywords: Event-interval sequence, distance measure, Dynamic Time Warping, Hungarian algorithm.

1 Introduction

Sequences of temporal intervals exist in many application domains, such as human motion databases, sign language, human activity monitoring, and medicine. Their main advantage over traditional sequences, which model series of instantaneous events, is that they incorporate the notion of duration in their event representation scheme. Due to this, they are used in a broad range of fields such as geo-informatics [29], cognitive science [4], linguistic analysis [5], music [24], and medicine [12]. Essentially, a sequence of event-intervals corresponds to a collection of labelled events accompanied by their start and end time values. We will call such sequence, *event-interval sequence*, and each labelled interval, *event-interval*. An example of such sequence—containing five event-intervals—is shown in Figure 1.

So far, most studies on event-interval sequences have been focusing on the aspect of knowledge discovery, such as mining patterns and association rules

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 229–244, 2011.

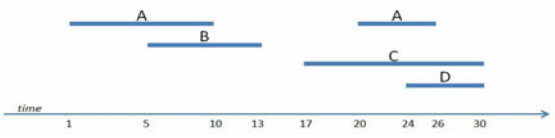


Fig. 1. An example of a sequence of interval-based events. This representation corresponds to the following sequence of event-intervals: $\mathcal{S} = \{(A, 1, 10), (B, 5, 13), (C, 17, 30), (A, 20, 26), (D, 24, 30)\}$.

that might be of interest to domain experts [2,11]. Surprisingly, very limited attention has been given on assessing the similarity of event-interval sequences [13]. Robust similarity measures for comparing such sequences would enable the utilization of existing clustering and classification methods, introduce new index structures for similarity search in event-interval sequence repositories, and would facilitate the implementation of recommendation systems and assistive applications.

Fig. 2. Two different sequences of interval-based events where the mapping to a sequence of instantaneous events produces the same representation for both

Existing similarity measures on symbolic sequences or time series are not directly applicable to event-interval sequences [13]. One could, for example, convert a sequence of event-intervals to a sequence of instantaneous events by only considering the start and end points of each event interval, and associating each of the two points with the same event label. This would result in a simplification of the representation of these sequences as they would be mapped to traditional sequences of instantaneous events. Thus, the solution to the problem would reduce to applying an existing distance/similarity measure for sequence matching, such as edit distance [16]. Nonetheless, this solution suffers from several shortcomings. Firstly, the size of the sequences and the alphabet (i.e., set of all possible event labels) would double since each event-interval label would be mapped to two instantaneous event labels. Secondly, crucial information about the pairwise temporal relations between the event-intervals in the sequence will be lost. Consider the example shown in Figure 2, where each sequences consists of two event-intervals with the same label. Obviously, the mapping for both sequences is the same, i.e., $\{A_{start}, A_{start}, A_{end}, A_{end}\}$. The relation between the two event intervals, however, is different. Hence, we can deduce that in order to provide a robust similarity measure for such sequences, their representation should include additional information about the relations between the event-intervals.

Our Contributions. In this paper, we formulate the problem of comparing sequences of event-intervals and show that solving the problem by directly mapping it to string matching fails to capture temporal dependencies between the interval-based events. In addition, we propose two distance measures to solve this problem: the first one maps the event-interval sequence to a sequence of vectors that can also be seen as a multi-dimensional time series. The second one attempts to identify correspondence between intervals of the two event-interval sequences, by taking into account the temporal relations that may occur between the events, and then employing minimum-weight bipartite matching to infer a similarity score for the sequences. Moreover, we propose a lower bound for the second method that can achieve significant speed-ups during similarity search. Finally, we present an extensive experimental evaluation on real datasets from three different domains. The methods are benchmarked with respect to their robustness to noise, nearest neighbor classification accuracy, and scalability. In addition, we study the pruning power and tightness of the proposed lower bound.

2 Event-Interval Sequences

Sequences of interval-based events can be represented in many ways. In this paper, we use the *e-sequence* [26,27] representation, which explicitly considers the start and end time stamps of each event.

Definition 1. ***(E-Sequence)*** *Given an alphabet σ of event labels, an* event-interval sequence *or* e-sequence $\mathcal{S}=\{S_1, \ldots, S_n\}$ *is an ordered set of events occurring over time intervals. Each $S_i = (E, t_{start}, t_{end})$ is called* event-interval, *where $S_i.E \in \sigma$, and $S_i.t_{start}$, $S_i.t_{end}$ denote the start and end time of $S_i.E$, respectively. Note that, we use $S_i.X$ to denote element X of S_i.*

Note that an event-interval corresponds to an event that occurs over a time interval. The temporal order of the event-intervals in an e-sequence is ascending based on their start time and in the case of ties it is descending based on their end time. If ties still exist, alphabetical ordering is applied based on labels. An example of an e-sequence is shown in Figure 1, and it corresponds to:

$$\mathcal{S} = \{(A, 1, 10), (B, 5, 13), (C, 17, 30), (A, 20, 26), (D, 24, 30)\}.$$

For simplicity, in this paper, we are only interested in the types of temporal relations between event-intervals in an e-sequence, and not in the actual duration of each event-interval. Let $\mathcal{I} = \{r_1, \ldots, r_{|\mathcal{I}|}\}$ be the set of all legal temporal relations that can exist between any pair of event-intervals. Based on Allen's model for interval temporal logic [3], we define $\mathcal{I}$ by considering seven types of relations between two event-intervals: *meet*, *match*, *overlap*, *contain*, *left-contain*, *right-contain*, *follow*. These relations are shown in Figure 3 and are described in detail by Papapetrou et al. [27]. Hence, $\mathcal{I} = \{meet, match, overlap, contain, left\text{-}contain, right\text{-}contain, follow\}$.

According to our formulation the same event label can occur many times within an e-sequence. Effectively, this allows two event-intervals of the same

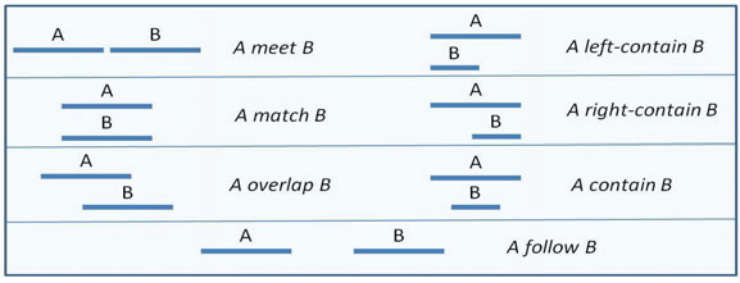


Fig. 3. The seven temporal relations between two event-intervals that are considered in this paper

label to overlap, which is a perfectly acceptable scenario, for example, in the case of nested for-loops in programming languages.

Using the above definitions, the problem studied in this paper can be formulated as follows:

Problem 1. **(E-sequence Distance)** Given two e-sequences $\mathcal{S}$ and $\mathcal{T}$, define a distance measure D, such that $\forall \mathcal{S}, \mathcal{T}$ it holds: $D(\mathcal{S}, \mathcal{T}) \geq 0$, $D(\mathcal{S}, \mathcal{S}) = 0$, and $D(\mathcal{S}, \mathcal{T}) = D(\mathcal{T}, \mathcal{S})$.

The degree to which $\mathcal{S}$ and $\mathcal{T}$ differ should be reflected in the value of $D(\mathcal{S}, \mathcal{T})$ and should be in accordance with the knowledge obtained from domain experts.

3 Distance Measures

In this section, we define two distance measures for comparing e-sequences and propose a lower bound for the second measure.

3.1 The Vector-Based DTW Distance

The first method employs a vector-based representation of e-sequences. For each e-sequence, a set of vectors is defined, where each vector indicates which and how many event-interval labels are active at specific time points in the e-sequence. The selected time points are all the instances in which an event-interval begins or ends.

Definition 2. ***(Event Vector)*** *Given an e-sequence $\mathcal{S}$ defined over an alphabet σ, an* event vector $V^t = (V_1^t, \ldots, V_{|\sigma|}^t)$ *consists of integer values, where each V_i^t records the number of occurrences of event E_i at time stamp t in $\mathcal{S}$.*

Hence, an e-sequence $\mathcal{S}$ can be mapped to an ordered set of event vectors $\mathcal{V}_\mathcal{S} = \{V^{t_0}, V^{t_1}, \ldots, V^{t_m}\}$. The set of time stamps $\{t_1, \ldots, t_m\}$ includes all time points in $\mathcal{S}$ where the "status" of at least one event-interval changes, i.e., an event-interval starts or ends. V^{t_0} is the null vector which denotes the initial condition at t_0 where no event takes place.

Example. Consider the e-sequence $\mathcal{S}$ shown in Figure 1. Given that $|\sigma| = 4$, the size of each event vector for $\mathcal{S}$ is also 4. The set of event vectors of $\mathcal{S}$ is defined as follows: $\mathcal{V}_\mathcal{S} = \{(0,0,0,0),(1,0,0,0),(1,1,0,0),(0,1,0,0),(0,0,0,0),(0,0,1,0),(1,0,1,0),(1,0,1,1),(0,0,1,1),(0,0,0,0)\}$.

Given two e-sequences $\mathcal{S}$ and $\mathcal{T}$, their vectors can also be seen as $|\sigma|$-dimension al time series; thus, their distance can be computed using Dynamic Time Warping (DTW) [14]. Vectors are compared using the $\mathcal{L}_1$ norm.

Despite the simplicity of this method, it only takes into account the temporal ordering of the event-intervals but does not explicitly consider any temporal relations. This may cause ambiguities in the representation by mapping two different e-sequences to the same set of event vectors.

Complexity. The sets of event vectors $\mathcal{V}_\mathcal{S}$ and $\mathcal{V}_\mathcal{T}$ can be computed in linear time. The time complexity of this DTW computation is $O(|\mathcal{V}_\mathcal{S}||\mathcal{V}_\mathcal{T}||\sigma|)$. Significant speedups in multidimensional time series similarity search under DTW can be achieved using existing lower bounding techniques [31].

3.2 `Artemis`: A Bipartite-Based Matching Distance

The second method, called `Artemis`is based on determining *correspondence* between pairs of event-intervals to infer the overall similarity of e-sequences. Given two e-sequences, correspondence is determined by the fraction of common relations between event-intervals in the e-sequences. The overall similarity score is derived from the sum of pairwise scores using the Hungarian algorithm (also known as Kuhn-Munkres algorithm) [23]. `Artemis` consists of two main steps: (a) the mapping step and (b) the matching step.

The Mapping Step. The first step of `Artemis` is to map each e-sequence $\mathcal{S}$ to an ordered set of temporal relations between event-intervals. Given the set of legal temporal relations between event-intervals $\mathcal{I}$ and two event-intervals S_i and S_j, $r(S_i, S_j)$ is the *event-interval relation* between S_i and S_j, with $r(S_i, S_j) \in \mathcal{I}$.

More specifically, for each event-interval $S_i \in \mathcal{S}$ we record the set of relations of S_i with $S_j \in \mathcal{S}, \forall j \neq i$. Three sets of relations are computed:

- $r_L(S_i) = \{r(S_j, S_i)|1 \leq j < i\}$: which contains the temporal relations of S_i with all event-intervals located on the left side of S_i in $\mathcal{S}$,
- $r_R(S_i) = \{r(S_i, S_j)|i < j \leq |S|\}$: which contains the temporal relations of S_i with all event-intervals located on the right side of S_i in $\mathcal{S}$, and
- $r_\varnothing(S_i) = \{r(\varnothing, S_i)\}$: which is a singleton with a *follow* relation between $\varnothing$—an extra symbol such that $\varnothing \notin \sigma$—and S_i.

In addition, let $r_{\varnothing L}(S_i) = r_\varnothing(S_i) \cup r_L(S_i)$. Note that symbol $\varnothing$ is introduced so that event labels are also taken into account: e-sequences that differ in event-interval relations but share similar event labels are assigned with smaller distance values than e-sequences that differ in both event labels and event-interval relations.

The Matching Step. Given two e-sequences $\mathcal{S}$ and $\mathcal{T}$, the matching step of `Artemis` computes a distance value for each pair of event-intervals $S_i \in \mathcal{S}$ and

$T_j \in \mathcal{T}$. The key idea behind this computation is to count the number of common relations between $r_{\varnothing L}(S_i)$ and $r_{\varnothing L}(T_j)$—the relations of S_i and T_j with all event-intervals located on their left side, including $\varnothing$—and the common relations between $r_R(S_i)$ and $r_R(T_j)$—the corresponding relations on their right side.

More formally, the *event-interval distance*, denoted as d_m, between two event-intervals $S_i \in \mathcal{S}$ and $T_j \in \mathcal{T}$ is defined as follows:

$$d_m(S_i, T_j) = \begin{cases} \dfrac{\max\{|\mathcal{S}|, |\mathcal{T}|\} - |r_{\varnothing L}(S_i) \cap r_{\varnothing L}(T_j)| - |r_R(S_i) \cap r_R(T_j)|}{\max\{|\mathcal{S}|, |\mathcal{T}|\}} & \text{, if } S_i.E = T_j.E \\ 1 & \text{, if } S_i.E \neq T_j.E \end{cases}$$

Let $D_{\mathcal{S},\mathcal{T}}$ be an $|\mathcal{S}| \times |\mathcal{T}|$ matrix, with $D(i,j) = d_m(S_i, T_j)$, $S_i \in \mathcal{S}$ and $T_j \in \mathcal{T}$. We call $D_{\mathcal{S},\mathcal{T}}$ the *event-interval distance matrix* of $\mathcal{S}$ and $\mathcal{T}$. Problem 1 reduces to the following optimization problem:

Problem 2. **(The Assignment Problem)** Given $\mathcal{S}$, $\mathcal{T}$, and $D_{\mathcal{S},\mathcal{T}}$, assign each event-interval in $\mathcal{S}$ to exactly one event-interval in $\mathcal{T}$ so that the total assignment cost is minimized.

Problem 2 can be solved by the Hungarian algorithm. Let the output of the algorithm be the following set $\mathcal{H}(\mathcal{S},\mathcal{T}) = \{h(S_1), \ldots, h(S_{|\mathcal{S}|}\}$ with an assignment cost $C(\mathcal{S},\mathcal{T})$. Each $h(S_i) \in \mathcal{H}(\mathcal{S},\mathcal{T})$ denotes the event-interval in $\mathcal{T}$ that $S_i \in \mathcal{S}$ is matched to by the Hungarian algorithm. The assignment cost $C(\mathcal{S},\mathcal{T})$ corresponds to the distance—called *Artemis Distance*—between $\mathcal{S}$ and $\mathcal{T}$.

Definition 3. ***(Artemis Distance)*** *Given $\mathcal{S}$, $\mathcal{T}$, $D_{\mathcal{S},\mathcal{T}}$, and $\mathcal{H}(\mathcal{S},\mathcal{T})$, the* `Artemis` *distance of $\mathcal{S}$ and $\mathcal{T}$ is defined as follows:*

$$\texttt{Artemis}(\mathcal{S},\mathcal{T}) = \sum_{i=1}^{\max\{|\mathcal{S}|,|\mathcal{T}|\}} D(S_i, h(S_i)),\ S_i \in \mathcal{S},\ h(S_i) \in \mathcal{H}(\mathcal{S},\mathcal{T}). \quad (1)$$

To handle the case of e-sequences of different size, "dummy" event-intervals are added with distance 1 *from all other event-intervals.*

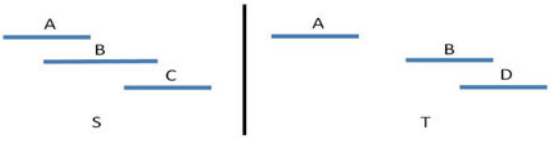


Fig. 4. Two e-sequences $\mathcal{S}$ and $\mathcal{T}$ used as an example for `Artemis`

Example. Figure 4 shows two e-sequences $\mathcal{S}$ and $\mathcal{T}$. At the *mapping step*, for each event-interval $S_i \in \mathcal{S}$ we compute $r_{\varnothing L}(A) = \{follow(\varnothing, A)\}$, $r_R(A) = \{overlap(A,B),\ follow(A,C)\}$, $r_{\varnothing L}(B) = \{follow(\varnothing, B),\ overlap(A,B)\}$, $r_R(B) = \{overlap(B,C)\}$, and $r_{\varnothing L}(C) = \{follow(\varnothing, C),\ follow(A,C),\ overlap(B,C)\}$.

For $\mathcal{T}$, we compute $r_{\varnothing L}(A) = \{follow(\varnothing, A)\}, r_R(A) = \{follow(A, B),\ follow(A, D)\}, r_{\varnothing L}(B) = \{follow(\varnothing, B),\ follow(A, B)\}, r_R(B) = \{overlap(B, D)\}$, and $r_{\varnothing L}(D) = \{follow(\varnothing, D),\ follow(A, D),\ overlap(B, D)\}$. At the *matching step*, the Hungarian algorithm gives $\mathcal{H}(\mathcal{S}, \mathcal{T}) = \{A, B, D\}$. Finally, $Artemis(\mathcal{S}, \mathcal{T}) = (2/3 + 2/3 + 1) = 7/3$.

Complexity. Let $m = max(|\mathcal{S}|, |\mathcal{T}|)$. Then, at the mapping step, $O(m^2)$ relations are enumerated, while the complexity of computing $D(\mathcal{S}, \mathcal{T})$ using hash-tables is $O(m^3)$. The cost of applying the cubic Hungarian algorithm to the two event-interval relation sets results to a total time complexity of $O(m^3)$. A lower bound for speeding up the computation of `Artemis` is described next.

3.3 Lower Bounding `Artemis`

The proposed lower bound can be computed in linear time and is based on the comparison of event label counts. By knowing the number of labels in which two e-sequences differ, we can determine a lower bound for their `Artemis` distance.

Given an e-sequences $\mathcal{S}$, we define an $|\sigma|$-dimensional vector $v^{\mathcal{S}}$, that stores, for each event label in σ, the count of event-intervals in $\mathcal{S}$ that share that label.

Theorem 1. *Given $\mathcal{S}$ and $\mathcal{T}$, the lower bound of $Artemis(\mathcal{S}, \mathcal{T})$ is defined as*

$$Artemis_{LB}(\mathcal{S}, \mathcal{T}) = \frac{k}{2} + \left(m - \frac{k}{2}\right)\left(\frac{k}{2m}\right) = k - \frac{k^2}{4m}, \tag{2}$$

where $k = ||v^{\mathcal{S}} - v^{\mathcal{T}}||_1$ and $m = max(|\mathcal{S}|, |\mathcal{T}|)$.

Proof. Knowing that $||v^{\mathcal{P}} - v^{\mathcal{Q}}||_1 = k$ we can be sure that the distance is at least $k/2$; those k event-intervals are matched with each other giving a score of 1 for each of the $k/2$ pairs. If $k/2$ is equal to the length of the e-sequences, then it is also their distance. However, if the differences refer to only a subset of all event-intervals, then these differences are reflected in the matching scores of the rest of the event-intervals. So, given that $m = max(|\mathcal{S}|, |\mathcal{T}|)$ and $m > \frac{k}{2}$, the rest of the $m - k/2$ event-intervals would have at least $k/2$ non-common relations. Thus, yielding an additional distance of $(m - k/2) \cdot (k/2m)$.

The proposed lower bound focuses on label counts and not on relations of event-intervals. When the differences of two e-sequences are restricted to event-interval labels, the lower bound is equal to the distance obtained by *Artemis*. On the other hand, when the e-sequences share the same event labels and differ only in the type of event-interval relations, then the lower bound yields zero score. The tightness and pruning power of the lower bound is studied on three datasets in Section 4.

4 Experiments

The performance of the proposed methods has been benchmarked on three real datasets. We studied their robustness to noise, classification accuracy, and scalability.

4.1 Experimental Setup

The proposed methods have been benchmarked on three real datasets (Table 1):

- ***Hepatitis*** [28]. The dataset contains information about 498 patients who have either Hepatitis B or Hepatitis C. The intervals represent the results of 25 regular tests.
- ***ASL*** [27]. The dataset contains 873 transcriptions from videos of American Sign Language expressions provided by Boston University. Each e-sequence in the dataset is one utterance expressed using American Sign Language.
- ***Pioneer*** [22]. This dataset was constructed from the Pioneer-1 dataset available in the UCI repository[1]. It contains time series sensor readings of the Pioneer-1 mobile robot. The data is segmented into event-intervals in which the robot takes action for some period of time.

The proposed methods were evaluated with respect to robustness against two types of artificial noise, k-NN classification accuracy, and scalability. In addition, the efficiency of the lower bound was tested by computing its *tightness* and its *pruning power* against 1-NN queries.

Table 1. Dataset Statistics

Dataset	# of e-sequences	# of intervals	e-sequence size			# of labels	# of classes
			min.	max.	average		
ASL	873	15675	4	41	18	216	5
Hepatitis	498	53921	15	592	108	147	2
Pioneer	160	8949	36	89	56	92	3

Robustness. The robustness of the proposed methods was tested on two types of artificial noise: *shifts* and *swaps*. In the first case, each interval within an e-sequence is shifted, with a certain *shift probability* value p, by an offset back or forth in time. Given a *distortion level* d as a percentage of the length of the whole sequence, a random value under the uniform distribution is chosen in that integer interval to determine the offset. An event-interval has equal probability to be shifted either back or forth in time, while interval durations remain unaffected. This type of artificial noise attempts to simulate noisy sources or recording devices. Real-world cases for this could include humans who learn sign language or who rehabilitate from brain injuries. Both p and d were set between 0.2 and 1, with step 0.2.

One drawback of employing such artificial noise is that shifting intervals could result to semantic invalidity, e.g., "robot walks forward" overlaps with "robot walks backwards". To avoid such cases, we further experiment with artificial noise which is based on *swaps* of event-interval labels, while the durations and relations of the event-intervals remain unaffected. Given an event-interval, the

[1] http://archive.ics.uci.edu/ml/

swap probability parameter determines if its label is swapped with the label of another event-interval which is chosen uniformly at random from the whole e-sequence. The swap probability parameter values ranged between 0.2 and 1 with step 0.2. Note that, noise is added off-line and that both methods were tested on exactly the same distorted sequences; to rule out score differences caused by the randomness factor.

Using the above two techniques, noise is inserted into e-sequences which then serve as the queries for 1-NN search. Given a database of e-sequences, a copy of an e-sequence is distorted, based on the parameter values, and then its nearest neighbor is found by scanning the database. This is performed for each e-sequence in the database. Ideally, we would like each noisy e-sequence (query) to be matched to the e-sequence from which it originated. We compared the DTW vector-based measure and `Artemis` in terms of: *retrieval accuracy* (the fraction of noisy queries for which the originating e-sequence is retrieved) and *rank of nearest neighbor* (for each query, the number of database e-sequences with distance less than or equal to that of the originating counterpart).

Classification. For the k-NN classification experiments, we studied the efficiency of the proposed methods under 1-NN and 3-NN classifiers. Due to the fact that the e-sequences in the *ASL* dataset can belong to up to 5 classes (wh-question, etc) simultaneously, and some elements do not have any label (simple affirmative phrases), we had to modify the algorithm. The *ASL* classifier returns a score in $[0, 1]$ denoting the average ratio of common class-labels with its k nearest neighbors. For example, a phrase with labels $\{a\}$ and neighbors with label sets $\{a\}$, $\{a, b\}$, and $\varnothing$ respectively, would yield $(1 + 0.5 + 0)/3$. For the *ASL* dataset, the result is the average sum.

We designed an additional experiment specifically for the *ASL* dataset. There exist in total 288 interval sequences in the dataset which portray exactly the same phrase in English with one or more other sequences. We studied what ratio of the 288 equivalent sequences was retrieved by examining the k nearest neighbors of every sample, for all possible values of k. Two values were monitored: the first corresponds to the ratio of the 288 sequences for which an identical phrase can be detected within their k nearest neighbors; the second is the ratio of the total sum of identical phrases detected within the k-nearest neighbors over the total number of pairs of identical labels. The two values would be the same if any phrase in English were to be represented only by at most two e-sequences in the dataset. This experiment was designed to investigate the suitability of the methods in the field of sign language and relevant application domains.

Scalability. In the attempt to evaluate the scalability of the proposed algorithms, we embedded time-monitoring functions in our implementations. For each e-sequence, we counted the time needed to map the e-sequence to the appropriate representation (set of event vectors for DTW and event-interval relation set for `Artemis`) and then to compare it against the whole database. The dataset, serving as the database, had already been transformed to the appropriate form. For the case of `Artemis`, we did not implement the use of hash-tables, thus the complexity is $O(n^4)$ w.r.t. the size of the e-sequences. Our experiments

were implemented in Java and were performed on a PC running Ubuntu Linux, equipped with Intel Core 2 Duo 2GHz CPU and 4GB RAM.

Lower Bounding. To assess the quality of the lower bound, we computed its pruning power for 1-NN queries in the database and its tightness. The pruning power is defined as the ratio of the number of pruned e-sequences, using $Artemis_{LB}$, over the total number of comparisons that would have been required if the database were to be scanned sequentially. The tightness is defined as the average ratio of the lower bound distance over the distance given by `Artemis`.

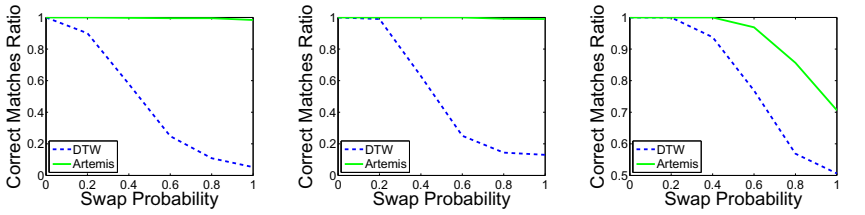
4.2 Results

Robustness. Testing the two measures for robustness against noise, `Artemis` was a clear winner for both types of noise insertion. The DTW vector-based approach displayed a significant decline in performance with the increase in the value of the probability and distortion level parameters. `Artemis` was able to maintain a very high rate of finding the noisy queries' originating counterparts. For almost all the datasets and experiments its retrieval accuracy was over 98%. These observations are shown in Figure 6, for the case of *shifts*, and in Figure 5, for the case of *swaps*. `Artemis` displayed a decline in performance only for *Pioneer* and the case of swaps. Note that in Figure 6 we do not show the performance of `Artemis` for *Pioneer* with respect to retrieval accuracy vs. the noise parameters since it always gave a retrieval accuracy of 100%.

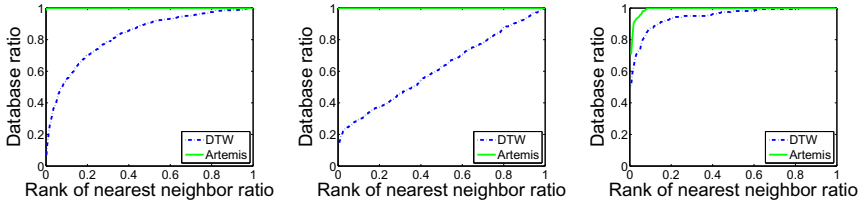
Classification. In Table 2 we see the results of the k-NN classification experiment. For the *Hepatitis* and *ASL*, the DTW vector-based approach performs marginally better, while for *Pioneer* `Artemis` is superior. Figure 7 depicts the results of the additional classification experiment conducted on *ASL*, in which the k nearest neighbors of each e-sequence are scanned to determine if they correspond to the same phrase in English. Figure 7a shows the ratio (of the e-sequences which do have equivalents) for which at least one identical phrase can be found within the k nearest neighbors. Figure 7b shows the ratio of the total sum of existing identical phrases that have been found. For both cases, `Artemis` gives a better grouping of semantically related e-sequences. Furthermore, empirical results showed that the groupings provided by `Artemis` are more meaningful for humans. Thus, `Artemis` proves more suitable for applications related to sign language.

Table 2. k-NN classification results, for $k = 1, 3$

Dataset	`Artemis` 1-NN	`Artemis` 3-NN	DTW 1-NN	DTW 3-NN
HepData	0.7209	0.7811	0.7403	0.8072
Pioneer	0.9750	0.9750	0.9375	0.9375
ASL	0.4307	0.4013	0.4358	0.4188



(a) *ASL*: retrieval accuracy for $p = 1.0$ **(b)** *Hepatitis*: retrieval accuracy for $p = 1.0$ **(c)** *Pioneer*: retrieval accuracy for $p = 1.0$



(d) *ASL*: ranks of NN for swap probability 1.0 **(e)** *Hepatitis*: ranks of NN for swap probability 1.0 **(f)** *Pioneer*: ranks of NN for swap probability 1.0

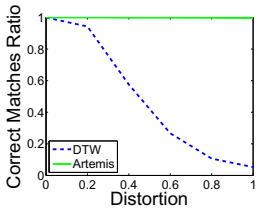
Fig. 5. Robustness experiment when noise is added in the form of swaps

Lower Bounding. Table 3 summarizes the assessment of $Artemis_{LB}$. The second column shows the tightness of the lower bound while the third shows the pruning power. The higher values are observed on *ASL*, contrary to *Pioneer* which yields the lowest scores. The latter consists of unequally represented classes; one of the three classes contains 102 out of 160 samples in total. Thus, its e-sequences are highly homogeneous and thereby render the lower bound technique unable to prune many of them.

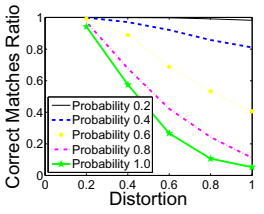
Table 3. Tightness and pruning power of $Artemis_{LB}$ for the 1-NN classification experiment and for the three datasets

Dataset	Tightness	1-NN pruning power
ASL	0.8837	0.7931
Hepatitis	0.7166	0.7012
Pioneer	0.6189	0.4855

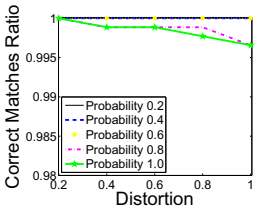
Scalability. The results of the scalability experiment can be seen in Figure 8. For *Pioneer* we observe that `Artemis` is slower than the DTW vector-based approach, as the complexity analysis of Section 3 would suggest. The runtime of `Artemis` blows up in the case of *Hepatitis*. For e-sequences of large size, `Artemis`' $O(n^4)$ complexity (without the use of hash-tables) significantly affects its performance. On the other hand, we observe that for *ASL* `Artemis` is faster than



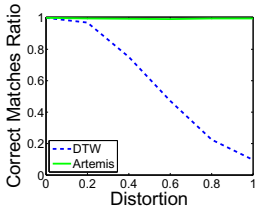
(a) *ASL*: retrieval accuracy for $p = 1.0$



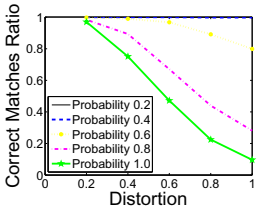
(b) *ASL*: DTW retrieval accuracy



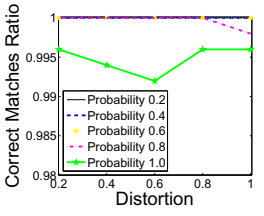
(c) *ASL*: `Artemis` retrieval accuracy



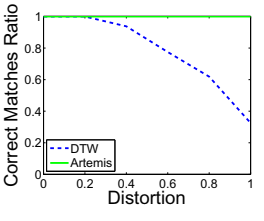
(d) *Hepatitis*: retrieval accuracy for $p = 1.0$



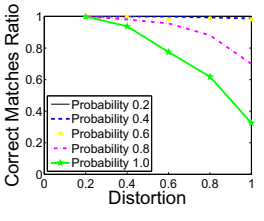
(e) *Hepatitis*: DTW retrieval accuracy



(f) *Hepatitis*: `Artemis` retrieval accuracy



(g) *Pioneer*: retrieval accuracy: $p = 1.0$



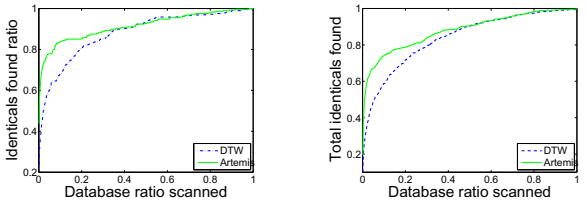
(h) *Pioneer*:DTW Retrieval Accuracy

Fig. 6. Robustness experiment when noise is added in the form of shifts

DTW. The reason for that is the large alphabet size, $|\sigma|$. A way to overcome this would be to prune the alphabet during the comparisons and keep only the union of the labels present in the compared e-sequences $\mathcal{S}, \mathcal{T}$. The alphabet size would then be at most $(|\mathcal{V}_\mathcal{S}| + |\mathcal{V}_\mathcal{T}|)$, and the new complexity of DTW $O(|\mathcal{V}_\mathcal{S}||\mathcal{V}_\mathcal{T}| \cdot (|\mathcal{V}_\mathcal{S}| + |\mathcal{V}_\mathcal{T}|))$.

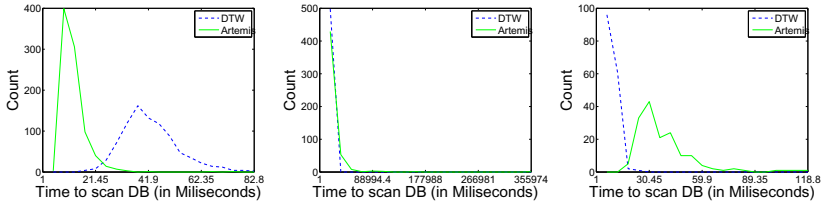
4.3 Lessons Learned

There was no clear winner, between the two proposed methods, in our k-NN classification experiments. The insight we acquired suggests that the choice must be application dependant. The additional experiment on *ASL* supports the thesis that, if semantic information resides in the relations between intervals, `Artemis` has an advantage over the DTW approach. This is further supported by the results of the noise-robustness experiments. In particular, the



(a) Identical phrases found **(b)** Total sum of identical phrases found

Fig. 7. Identical phrase experiment: *ASL*



(a) *ASL* **(b)** *Hepatitis* **(c)** *Pioneer*

Fig. 8. Histograms of the time (in milliseconds) required to compare each element against the whole dataset

DTW vector-based approach displayed a deterioration in performance in accordance with the increase of noise. `Artemis` proved highly robust against the types of noise that we experimented with. The advantage of `Artemis` is that a correspondence amongst event-intervals is determined based on the relations of the event-intervals within the e-sequences; contrary to the DTW approach which examines the e-sequences point-by-point and out of their context. The correspondence among event-intervals allows `Artemis` to easily identify the originating counterpart of noisy e-sequences; e-sequences with the same count of each event label yield in the majority of cases lower distance scores than others with different event labels and size.

The scalability experiments showed that DTW outperforms `Artemis`, which suffers from a computational blow up for large e-sequences. On the other hand, DTW becomes slower when the alphabet size is significantly larger than the size of the e-sequences. To speed up search using `Artemis`, our lower bound technique proved significantly tight; the average tightness was measured at 61.8% for *Pioneer*, and up to 88% for *ASL*. This translates to a pruning power of 48.5% to 79.3% over the brute force sequential scan of the database.

5 Related Work

Existing work on interval-based sequences has so far been focusing merely on frequent pattern and association rule mining.

Several approaches [17,30] consider discovering frequent intervals in databases, where intervals appear sequentially and are not labelled, while others [7] consider temporally annotated sequential patterns where transitions from one event to another have a time duration. A graph-based approach [10] represents each temporal pattern by a graph considering only two types of relations between events (*follow* and *overlap*). An approach for mining sequences of interval-based events in a database is discussed in Fu et al. [11], however it is limited to certain forms of patterns.

A generalized interval-based framework [15] improves support counting techniques for mining interval-based episodes; nonetheless, no temporal relations are considered between events. Apriori-based techniques [8,9,19,6,1] for finding temporal patterns and association rules on interval-based event sequences have been proposed, some [9] also applying interestingness measures to evaluate the significance of the findings.

BFS-based and DFS-based approaches [25,26,27,32] apply efficient pruning techniques, thus reducing the inherent exponential complexity of the mining problem, while a non-ambiguous event-interval representation is defined [33] that considers start and end points of e-sequences and converts them to a sequential representation. Finally, there has been some recent work on mining semi-partial orders of time intervals [22].

Moreover, in Ale et. al [2], the lifetime of an item is defined as the time between the first and the last occurrence and the temporal support is calculated with respect to this interval. Finally, Lu et. al [18] study inter-transaction association rules by merging all itemsets within a sliding time window inside a transaction.

Recent work on *margin-closed* patterns [21,22], focuses on significantly reducing the number of reported patterns by favouring longer patterns and suppressing shorter patterns with similar frequencies. The extracted set of margin-closed patterns may include a significantly smaller set of patterns compared to the set of closed patterns while retaining the most important information about the data. A unifying view of temporal concepts and data models has been formulated [20] to enable categorization of existing approaches for unsupervised pattern mining from symbolic temporal data; Time point-based methods and interval-based methods as well as univariate and multivariate methods are considered.

6 Summary and Conclusions

We have defined the problem of comparing sequences of interval-based events and presented two polynomial-time algorithms. The first reduces the problem to matching vectors, while the second, `Artemis`, is based on determining correspondence among intervals and maps the problem to the assignment problem. The two measures were tested for their robustness against two types of artificial noise, their k-NN classification power, and, finally, their scalability. `Artemis` demonstrates a remarkable robustness against noise, while there exist cases for both measures in which they perform faster than the other. Additionally, we developed a linear-time lower-bounding technique for `Artemis`, which we tested in terms of tightness and pruning power.

Directions for future work include studying the applicability of the proposed methods to other application domains and real-world scenarios. Furthermore, we plan on investigating alternative problems to assess the similarity of event-interval sequences, such as the Maximum Common Subsequence and the Maximum Contiguous Subsequence. Devising algorithms for these problems would allow the direct application of event-interval sequences to domains such as anomaly detection, profiling of executable files, network monitoring, and many others.

Acknowledgements. This work was supported in part by the Academy of Finland ALGODAN Centre of Excellence. We would like to thank D. Patel for providing us with the *Hepatitis* dataset, and F. Mörchen for the *Pioneer* dataset.

References

1. Abraham, T., Roddick, J.F.: Incremental meta-mining from large temporal data sets. In: Proceedings of the Workshops on Data Warehousing and Data Mining, pp. 41–54 (1999)
2. Ale, J.M., Rossi, G.H.: An approach to discovering temporal association rules. In: Proc. of the SAC, pp. 294–300 (2000)
3. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 832–843 (1983)
4. Berendt, B.: Explaining preferred mental models in Allen inferences with a metrical model of imagery. In: Proceedings of the Annual Conference of the Cognitive Science Society, pp. 489–494 (1996)
5. Bergen, B., Chang, N.: Embodied construction grammar in simulation-based language understanding. In: Construction grammars: Cognitive grounding and theoretical extensions, pp. 147–190 (2005)
6. Chen, X., Petrounias, I.: Mining temporal features in association rules. In: Żytkow, J.M., Rauch, J. (eds.) PKDD 1999. LNCS (LNAI), vol. 1704, pp. 295–300. Springer, Heidelberg (1999)
7. Giannotti, F., Nanni, M., Pedreschi, D.: Efficient mining of temporally annotated sequences. In: SDM, vol. 6, pp. 346–357 (2006)
8. Höppner, F.: Discovery of temporal patterns. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 192–203. Springer, Heidelberg (2001)
9. Höppner, F., Klawonn, F.: Finding informative rules in interval sequences. In: Hoffmann, F., Adams, N., Fisher, D., Guimarães, G., Hand, D.J. (eds.) IDA 2001. LNCS, vol. 2189, pp. 123–132. Springer, Heidelberg (2001)
10. Hwang, S.-Y., Wei, C.-P., Yang, W.-S.: Discovery of temporal patterns from process instances. Computers in Industry 53(3), 345–364 (2004)
11. Kam, P., Fu, A.W.: Discovering temporal patterns for interval-based events. In: Kambayashi, Y., Mohania, M., Tjoa, A.M. (eds.) DaWaK 2000. LNCS, vol. 1874, pp. 317–326. Springer, Heidelberg (2000)
12. Kosara, R., Miksch, S.: Visualizing complex notions of time. Studies in Health Technology and Informatics, 211–215 (2001)
13. Kostakis, O., Papapetrou, P., Hollmén, J.: Distance measure for querying arrangements of temporal intervals. In: Proc. of ACM Pervasive Technologies Related to Assistive Environments, PETRA (2011)

14. Kruskal, J.B., Liberman, M.: The symmetric time warping algorithm: From continuous to discrete. In: Time Warps. Addison-Wesley, Reading (1983)
15. Laxman, S., Sastry, P., Unnikrishnan, K.: Discovering frequent generalized episodes when events persist for different durations. IEEE Transactions on Knowledge and Data Engineering 19(9), 1188–1201 (2007)
16. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics 10(8), 707–710 (1966)
17. Lin, J.-L.: Mining maximal frequent intervals. In: Proc. of SAC, pp. 624–629 (2003)
18. Lu, H., Han, J., Feng, L.: Stock movement prediction and n-dimensional inter-transaction association rules. In: Proc. of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pp. 12:1–12:7 (1998)
19. Mooney, C., Roddick, J.F.: Mining relationships between interacting episodes. In: Proc. of SDM (2004)
20. Mörchen, F.: Unsupervised pattern mining from symbolic temporal data. SIGKDD Explor. Newsl. 9, 41–55 (2007)
21. Mörchen, F.: Temporal pattern mining in symbolic time point and time interval data. In: Proc. of ACM SIGKDD (2010)
22. Mörchen, F., Fradkin, D.: Robust mining of time intervals with semi-interval partial order patterns. In: SDM, pp. 315–326 (2010)
23. Munkres, J.: Algorithms for the assignment and transportation problems. Journal of the Society for Industrial and Applied Mathematics 5(1), 32–38 (1957)
24. Pachet, F., Ramalho, G., Carrive, J.: Representing temporal musical objects and reasoning in the MusES system. Journal of new music research 25(3), 252–275 (1996)
25. Papapetrou, P., Benson, G., Kollios, G.: Discovering frequent poly-regions in dna sequences. In: Proc. of the IEEE ICDM Workshop on Data Mining in Bioinformatics, pp. 94–98 (2006)
26. Papapetrou, P., Kollios, G., Sclaroff, S., Gunopulos, D.: Discovering frequent arrangements of temporal intervals. In: Proc. of IEEE ICDM, pp. 354–361 (2005)
27. Papapetrou, P., Kollios, G., Sclaroff, S., Gunopulos, D.: Mining frequent arrangements of temporal intervals. In: Knowledge and Information Systems (KAIS), vol. 21, pp. 133–171 (2009)
28. Patel, D., Hsu, W., Lee, M.: Mining relationships among interval-based events for classification. In: Proc. of ACM SIGMOD, pp. 393–404 (2008)
29. Pissinou, N., Radev, I., Makki, K.: Spatio-temporal modeling in video and multimedia geographic information systems. GeoInformatica 5(4), 375–409 (2001)
30. Villafane, R., Hua, K.A., Tran, D., Maulik, B.: Knowledge discovery from series of interval events. Intelligent Information Systems 15(1), 71–89 (2000)
31. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing multidimensional time-series. The VLDB Journal 15, 1–20 (2006)
32. Winarko, E., Roddick, J.F.: Armada - an algorithm for discovering richer relative temporal association rules from interval-based data. Data & Knowledge Engineering 63(1), 76–90 (2007)
33. Wu, S.-Y., Chen, Y.-L.: Mining nonambiguous temporal patterns for interval-based events. IEEE Transactions on Knowledge and Data Engineering 19(6), 742–758 (2007)

Unifying Guilt-by-Association Approaches: Theorems and Fast Algorithms

Danai Koutra[1], Tai-You Ke[2], U. Kang[1], Duen Horng (Polo) Chau[1], Hsing-Kuo Kenneth Pao[2], and Christos Faloutsos[1]

[1] School of Computer Science, Carnegie Mellon University
{danai,ukang,dchau,christos}@cs.cmu.edu
[2] Dept. of Computer Science & Information Engineering
National Taiwan Univ. of Science & Technology
{M9815071,pao}@mail.ntust.edu.tw

Abstract. If several friends of *Smith* have committed petty thefts, what would you say about *Smith*? Most people would not be surprised if *Smith* is a hardened criminal. **Guilt-by-association** methods combine weak signals to derive stronger ones, and have been extensively used for anomaly detection and classification in numerous settings (e.g., accounting fraud, cyber-security, calling-card fraud).

The focus of this paper is to compare and contrast several very successful, *guilt-by-association* methods: *Random Walk with Restarts*, Semi-Supervised Learning, and *Belief Propagation* (BP).

Our main contributions are two-fold: (a) theoretically, we prove that all the methods result in a similar matrix inversion problem; (b) for practical applications, we developed FaBP, a fast algorithm that yields 2× speedup, equal or higher accuracy than BP, and is guaranteed to converge. We demonstrate these benefits using synthetic and real datasets, including YahooWeb, one of the largest graphs ever studied with BP.

Keywords: Belief Propagation, Random Walk with Restart, Semi-Supervised Learning, probabilistic graphical models, inference.

1 Introduction

Network effects are very powerful, resulting even in popular proverbs like "birds of a feather flock together". In social networks, obese people tend to have obese friends [5], happy people tend to make their friends happy too [7], and in general, people usually associate with like-minded friends with respect to politics, hobbies, religion etc. Thus, knowing the types of a few nodes in a network, (say, "honest" vs "dishonest"), we would have good chances to guess the types of the rest of the nodes.

Informally, the guilt-by-association problem (or label propagation in graphs) is defined as follows:

Given: a graph with N nodes and M edges; n_+ and n_- nodes labeled as members of the positive and negative class respectively.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 245–260, 2011.

Find: the class memberships of the rest of the nodes, assuming that neighbors influence each other.

The influence can be "homophily", meaning that nearby nodes have similar labels, or "heterophily", meaning the reverse (e.g., talkative people tend to prefer silent friends, and vice-versa). Homophily appears in numerous settings, for example: (a) *Personalized PageRank*: if a user likes some pages, she would probably like other pages that are heavily connected to her favorites. (b) *Recommendation systems*: if a user likes some products (i.e., members of *positive* class), which other products should get *positive* scores? (c) *Accounting and calling-card fraud*: if a user is dishonest, his/her contacts are probably dishonest too.

There are several, closely related methods that address the homophily problem, and some - among which is our proposed FABP method, improved on *Belief Propagation* - that address both homophily *and* heterophily. We focus on three of them: Personalized PageRank (or "Personalized Random Walk with Restarts", or just RWR), Semi-Supervised Learning (SSL), and Belief Propagation (BP). How are these methods related? Are they identical? If not, which method gives the best accuracy? Which method has the best scalability?

These questions are exactly the focus of this work. In a nutshell, we contribute by answering the above questions, and providing a fast algorithm inspired by our theoretical analysis:

- *Theory & Correspondences*: the three methods are closely related, but not identical.
- *Algorithm & Convergence*: we propose FABP, a fast, accurate and scalable algorithm, and provide the conditions under which it converges.
- *Implementation & Experiments*: finally, we propose a HADOOP-based algorithm, that scales to *billion-node* graphs, and we report experiments on one of the largest graphs ever studied in the open literature. Our FABP method achieves about 2× better runtime.

2 Related Work

RWR, SSL and BP are very popular techniques, with numerous papers using or improving them. Here, we survey the related work for each method.

RWR is the method underlying Google's classic PageRank algorithm [2]. RWR's many variations include *Personalized PageRank* [10], *lazy random walks* [20], and more [24, 21]. Related methods for node-to-node distance (but not necessarily *guilt-by-association*) include Pegasus [15], parameterized by *escape probability* and *round-trip probability*.

According to conventional categorization, SSL approaches are classified into four categories [28]: *low-density separation* methods, *graph-based* methods, methods for *changing the representation*, and *co-training* methods. The principle behind SSL is that unlabeled data can help us decide the "metric" between data points and improve the models' performance. A very recent use of SSL for multi-class settings has been proposed in [12]. In this work, we mainly study the graph-based SSL methods.

BP [23], being an efficient inference algorithm on probabilistic graphical models, has been successfully applied to numerous domains, including error-correcting codes [16], stereo imaging in computer vision [6], fraud detection [19, 22], and malware detection[3]. Extensions of BP include *Generalized Belief Propagation* (GBP), that takes a multi-resolution view point, grouping nodes into regions [27]; however, how to construct good regions is still an open research problem. Thus, we focus on standard BP, which is better understood. Here, we study how the parameter choices for BP helps accelerate the algorithms, and how to implement the method on top of HADOOP [1] (open-source MapReduce implementation). This focus differentiates our work from existing research which speeds up BP by exploiting the graph structure [4, 22] or the order of message propagation [9].

*S*ummary: None of the above papers show the relationships between the three methods, or discuss the parameter choices (e.g., homophily factor). Table 1 qualitatively compares the methods. BP supports heterophily, but there is no guarantee on convergence. Our FABP algorithm improves on it to provide convergence.

Table 1. Qualitative comparison of 'guilt-by-association' (GBA) methods

GBA Method	Heterophily	Scalability	Convergence
RWR	No	Yes	Yes
SSL	No	Yes	Yes
BP	Yes	Yes	?
FABP	Yes	Yes	Yes

3 Theorems and Correspondences

In this section we present the three main formulas that show the similarity of the following methods: binary BP and specifically our proposed approximation, the linearized BP (FABP), Gaussian BP (GAUSSIANBP), Personalized RWR (RWR), and Semi-Supervised learning (SSL).

For the homophily case, all the above methods are similar in spirit, and closely related to diffusion processes: the n_+ nodes that belong to class "+" (say, "green"), act as if they taint their neighbors (diffusion) with green color, and similarly do the negative nodes with, say, red color. Depending on the strength of homophily, or equivalently the speed of diffusion of the color, eventually we have green-ish neighborhoods, red-ish neighborhoods, and bridge-nodes (half-red, half-green).

The solution vectors for each method obey very similar equations: they all involve a matrix inversion, where the matrix consists of a diagonal matrix and a weighted or normalized version of the adjacency matrix. Table 2 has the definitions of the symbols that are used in the following discussion, and Table 3 shows the resulting equations, carefully aligned to highlight the correspondences.

Table 2. Major Symbols and Definitions. (matrices in bold capital, vectors in bold lowercase, and scalars in plain font).

Symbols	Definitions	Explanations
n	number of nodes in the graph	
$\mathbf{A}$	$n \times n$ symmetric adjacency matrix	
$\mathbf{D}$	$n \times n$ diagonal matrix of degrees	$D_{ii} = \sum_j A_{ij}$ and $D_{ij} = 0$ for $i \neq j$
$\mathbf{I}$	$n \times n$ identity matrix	
$\mathbf{b_h}$	"about-half" final beliefs $\boldsymbol{b} - 0.5$	$\boldsymbol{b} = n \times 1$ vector of the BP final beliefs $b(i)\{> 0.5, < 0.5\}$ means $i \in \{$"+", "-"$\}$ class $b(i) = 0$ means i is unclassified (neutral)
$\boldsymbol{\phi_h}$	"about-half" prior beliefs, $\boldsymbol{\phi} - 0.5$	$\boldsymbol{\phi} = n \times 1$ vector of the BP prior beliefs
h_h	"about-half" homophily factor $h - 0.5$	$h = \psi($"+","+"$)$: BP propagation matrix entry $h \to 0$ means strong heterophily $h \to 1$ means strong homophily

Table 3. Main results, to illustrate correspondence: $n \times n$ matrices in bold capital, $n \times 1$ vectors in bold lowercase, and scalars in plain font

Method	matrix	unknown		known
RWR	$[\mathbf{I} \quad - c\mathbf{A}\mathbf{D}^{-1}]\times$	$\boldsymbol{x}$	$=$	$(1-c)\ \boldsymbol{y}$
SSL	$[\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]\times$	$\boldsymbol{x}$	$=$	$\boldsymbol{y}$
Gaussian BP = SSL	$[\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]\times$	$\boldsymbol{x}$	$=$	$\boldsymbol{y}$
FaBP	$[\mathbf{I} + a\mathbf{D} - c'\mathbf{A}]\times$	$\mathbf{b_h}$	$=$	$\boldsymbol{\phi_h}$

Theorem 1 (FaBP)**.** *The solution to Belief Propagation can be approximated by the linear system*

$$[\mathbf{I} + a\mathbf{D} - c'\mathbf{A}]\mathbf{b_h} = \boldsymbol{\phi_h} \tag{1}$$

where $a = 4h_h^2/(1 - 4h_h^2)$*, and* $c' = 2h_h/(1 - 4h_h^2)$*. The definitions of* h_h*,* $\phi_\mathbf{h}$ *and* $\mathbf{b_h}$ *are given in Table 2. Specifically,* $\phi_\mathbf{h}$ *corresponds to the prior beliefs of the nodes, and node* i*, about which we have no information, has* $\phi_h(i) = 0$*;* $\mathbf{b_h}$ *corresponds to the vector of our final beliefs for each node.*

Proof. The goal behind the "about-half" is the linearization of BP using Maclaurin expansions. The preliminary analysis of FaBP, and the necessary lemmas for the linearization of the original BP equations are given in Appendix A. For the detailed proof of this theorem see Appendix B. □

Lemma 1 (Personalized RWR)**.** *The linear system for RWR given an observation* $\boldsymbol{y}$*, is described by the following equation:*

$$[\mathbf{I} - c\mathbf{A}\mathbf{D}^{-1}]\boldsymbol{x} = (1 - c)\boldsymbol{y} \tag{2}$$

where $1 - c$ *is the restart probability,* $c \in [0, 1]$*. Similarly to the BP case above,* $\boldsymbol{y}$ *corresponds to the prior beliefs for each node, with the small difference that* $y_i = 0$ *means that we know nothing about node* i*, while a positive score* $y_i > 0$ *means that the node belongs to the positive class (with the corresponding strength).*

Proof. See [11], [24]. □

Lemma 2 (SSL and Gaussian BP)**.** *Suppose we are given l labeled nodes (x_i, y_i), $i = 1, \ldots, l$, $y_i \in \{0, 1\}$, and u unlabeled nodes $(x_{l+1}, ..., x_{l+u})$. The solution to a Gaussian BP and SSL problem is given by the linear system:*

$$[\alpha(\mathbf{D} - \mathbf{A}) + \mathbf{I}]\boldsymbol{x} = \boldsymbol{y} \tag{3}$$

where α is related to the coupling strength (homophily) of neighboring nodes, $\boldsymbol{y}$ represents the labels of the labeled nodes and, thus, it is related to the prior beliefs in BP, and $\boldsymbol{x}$ corresponds to the labels of all the nodes or equivalently the final beliefs in BP.

Proof. See Appendix B and [25], [28]. □

Lemma 3 (RWR-SSL correspondence)**.** *On a regular graph (i.e., all nodes have the same degree d), RWR and SSL can produce identical results if*

$$\alpha = \frac{c}{(1-c)d}\,. \tag{4}$$

That is, we need to align carefully the homophily strengths α and c.

Proof. See Appendix B. □

In an arbitrary graph the degrees are different, but we can still make the two methods give the same results if each node has a different α_i instead of α. Specifically, for node i with degree d_i, the quantity α_i should be $\frac{c}{(1-c)d_i}$. The following section illustrates the correspondence between RWR and SSL.

3.1 Arithmetic Examples

Here we illustrate that SSL and RWR result in closely related solutions. We study both the case with variable α_i for each node i, and the case with fixed $\alpha = c/((1-c)\bar{d})$, where $\bar{d}$ is the average degree.

We generated a random graph using the Erdős-Rényi model, $G(n, p) = G(100, 0.3)$. Figure 1 shows the scatter-plot: each node i has a corresponding blue circle $(x_{1i},\ y_{1i})$ for variable α_i, and also a red star $(x_{2i},\ y_{2i})$ for fixed α. The coordinates of the points are the RWR and SSL scores, respectively. Figure 1(b) shows a magnification of the central part of Fig. 1(a). Notice that the red stars (fixed α) are close to the 45-degree line, while the blue circles (variable α_i) are exactly on the 45-degree line. The conclusion is that (a) the SSL and RWR scores are similar, and (b) the rankings are the same: whichever node is labeled as "positive" by SSL, gets a high score by RWR, and conversely.

4 Analysis of Convergence

In this section we provide the sufficient, but not necessary conditions for which our method, FaBP, converges. The implementation details of FaBP are described in the upcoming Section 5. Lemmas 4, 5, and 8 give the convergence

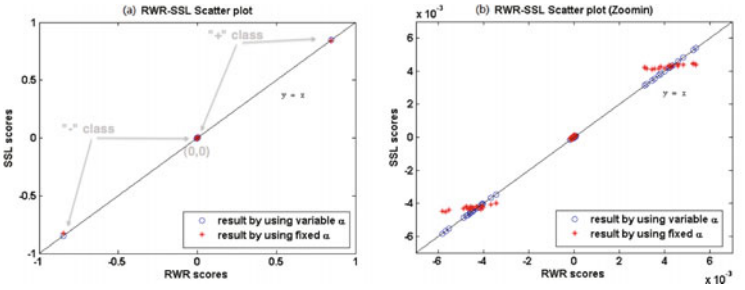


Fig. 1. Scatter plot showing the similarities between SSL and RWR. Scores of SSL and RWR for the nodes of a random graph: blue circles (perfect equality – variable α_i) and red stars (fixed α). The right figure is a zoom-in of the left. Most red stars are on or close to the diagonal: the two methods give similar scores, and identical assignments to positive/negative classes.

conditions. At this point we should mention that work on the convergence of a variant of BP, Gaussian BP, is done in [18] and [25]. The reasons that we focus on BP are that (a) it has a solid, Bayesian foundation, and (b) it is more general than the rest, being able to handle heterophily (as well as multiple-classes, that we don't elaborate here).

All our results are based on the power expansion required to compute the inverse of a matrix of the form $\mathbf{I} - \mathbf{W}$; all the methods undergo this process, as we show in Table 3. Specifically, we need the inverse of the matrix $\mathbf{I}+a\mathbf{D}-c'\mathbf{A} = \mathbf{I} - \mathbf{W}$, which is given by the expansion:

$$(\mathbf{I} - \mathbf{W})^{-1} = \mathbf{I} + \mathbf{W} + \mathbf{W}^2 + \mathbf{W}^3 + ... \tag{5}$$

and the solution of the linear system is given by the formula

$$(\mathbf{I} - \mathbf{W})^{-1}\phi_\mathbf{h} = \phi_\mathbf{h} + \mathbf{W} \cdot \phi_\mathbf{h} + \mathbf{W} \cdot (\mathbf{W} \cdot \phi_\mathbf{h}) + ... \tag{6}$$

This method, also referred to as the *Power Method*, is fast since the computation can be done in iterations, each one of which consists of a sparse-matrix/vector multiplication. In this section we examine its convergence conditions.

Lemma 4 (Largest eigenvalue)**.** *The series* $\sum_{k=0}^{\infty} |c'\mathbf{A} - a\mathbf{D}|^k$ *converges iff* $\lambda(\mathbf{W}) < 1$*, where* $\lambda(\mathbf{W})$ *is the magnitude of the largest eigenvalue of* $\mathbf{W}$*.*

Given that the computation of the largest eigenvalue is non-trivial, we suggest using one of the following lemmas, which give a closed form for computing the "about-half" homophily factor, h_h.

Lemma 5 (1-norm)**.** *The series* $\sum_{k=0}^{\infty} |c'\mathbf{A} - a\mathbf{D}|^k$ *converges if*

$$h_h < \frac{1}{2 + 2\max_j d_{jj}} \tag{7}$$

where d_{jj} *are the elements of the diagonal matrix D.*

Proof. The proof is based on the fact that the power series converges if the 1-norm, or equivalently the ∞-norm, of the symmetric matrix $\mathbf{W}$ is smaller than 1. The detailed proof is shown in Appendix C. □

Lemma 6 (Frobenius norm)**.** *The series* $\sum_{k=0}^{\infty} |c'\mathbf{A} - a\mathbf{D}|^k$ *converges if*

$$h_h < \sqrt{\frac{-c_1 + \sqrt{c_1^2 + 4c_2}}{8c_2}} \tag{8}$$

where $c_1 = 2 + \sum_i d_{ii}$ *and* $c_2 = \sum_i d_{ii}^2 - 1$.

Proof. This upper bound for h_h is obtained when we consider the Frobenius norm of matrix $\mathbf{W}$ and we solve the inequality $\| \mathbf{W} \|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} |\mathbf{W}_{ij}|^2} < 1$ with respect to h_h. We omit the detailed proof. □

Equation (8) is preferable over (7) when the degrees of the graph's nodes demonstrate considerable standard deviation. The 1-norm yields small h_h for very big highest degree, while the Frobenius norm gives a higher upper bound for h_h. Nevertheless, we should bear in mind that h_h should be a sufficiently small number in order for the "about-half" approximations to hold, because of the "about-half" approximations done in the analysis of FaBP.

5 Proposed Algorithm: FaBP

Based on the analysis in Sections 3 and 4, we propose the FaBP algorithm:

- **Step 1:** Pick h_h to achieve convergence: $h_h = \max\{(7), (8)\}$ and compute the parameters a and c' as described in Theorem 1.
- **Step 2:** Solve the linear system (1). Notice that all the quantities involved in this equation are close to zero.
- **Step 3** (optional): If the achieved accuracy is not sufficient, run a few iterations of BP using the values computed in Step 2 as the prior node beliefs.

In the datasets we studied, the optional step was not required, as FaBP achieved equal or higher accuracy than BP, while running in less time.

6 Experiments

We present experimental results to answer the following questions:

Q1: How accurate is FaBP?
Q2: Under what conditions does FaBP converge?
Q3: How sensitive is FaBP to the values of h and ϕ?
Q4: How does FaBP scale on very large graphs with billions of nodes and edges?

The graphs we used in our experiments are summarized in Table 4. To answer the first three questions, we used the DBLP dataset [8], which consists of 14,376 papers, 14,475 authors, 20 conferences, and 8,920 terms. Each paper is connected to its authors, the conference in which it appeared and the terms in its title. Only a small portion of the nodes are labeled: 4,057 authors, 100 papers, and all the conferences. We adapted the labels of the nodes to two classes: AI (Artificial Intelligence) and *not* AI (= Databases, Data Mining and Information Retrieval). In each trial, we ran FaBP on the DBLP network where $(1-p)\%$ of the labels of the papers and the authors had been discarded, with $p \in \{0.1\%, 0.2\%, 0.3\%, 0.4\%, 0.5\%, 5\%\}$. Then, we tested the classification accuracy on the nodes whose labels were discarded. To avoid combinatorial explosion in the presentation of the results, we consider $\{h_h, priors\} = \{0.002, \pm 0.001\}$ as the anchor values, and then, we vary one parameter at a time. When the results are the same for different values of $p\%$, due to lack of space, we randomly pick the plots to present.

To answer the last question, we used the YahooWeb dataset, as well as Kronecker graphs – synthetic graphs generated by the Kronecker generator [17]. YahooWeb is a Web graph containing 1.4 billion web pages and 6.6 billion edges; we automatically labeled 11 million educational and 11 million adult web pages. We used 90% of these labeled data to set the node priors, and the remaining 10% to evaluate the accuracy. For parameters, we set h_h to 0.001 using Lemma 6 (Frobenius norm), and the magnitude of the prior beliefs to ± 0.001.

Table 4. Order and size of graphs

Dataset	YahooWeb	Kronecker 1	Kronecker 2	Kronecker 3	Kronecker 4	DBLP
# nodes	1,413,511,390	177,147	120,552	59,049	19,683	37,791
# edges	6,636,600,779	1,977,149,596	1,145,744,786	282,416,200	40,333,924	170,794

6.1 Q1: Accuracy

Figure 2 shows the scatter plots of beliefs (FaBP vs BP) for each node of the DBLP data. We observe that FaBP and BP result in practically the same beliefs for all the nodes in the graph, when ran with the same parameters, and thus, they yield the same accuracy. Conclusions are identical for any labeled set-size we tried (0.1% and 0.3% shown in Fig. 2).

Observation 1. FaBP *and* BP *agree on the classification of the nodes when ran with the same parameters.*

6.2 Q2: Convergence

We examine how the value of the "about-half" homophily factor affects the convergence of FaBP. In Fig. 3 the red line annotated with "max $|eval| = 1$" splits the plots into two regions; (a) on the left, the Power Method converges

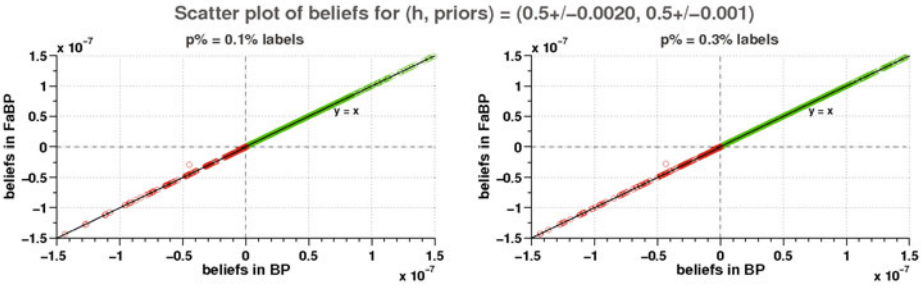


Fig. 2. The quality of scores of FABP is near-identical to BP, i.e. all the points are on the 45-degree line in the scatter plot of the DBLP sub-network node beliefs (FABP vs BP); red/green points correspond to nodes classified as "AI/not-AI" respectively

and FABP is accurate, (b) on the right, the Power Method diverges resulting in significant drop in the classification accuracy. We annotate the number of classified nodes for the values of h_h that leave some nodes unclassified due to numerical representation issues. The low accuracy scores for the smallest values of h_h are due to the unclassified nodes, which are counted as misclassifications. The Frobenius norm-based method yields greater upper bound for h_h than the 1-norm based method, preventing any numerical representation problems.

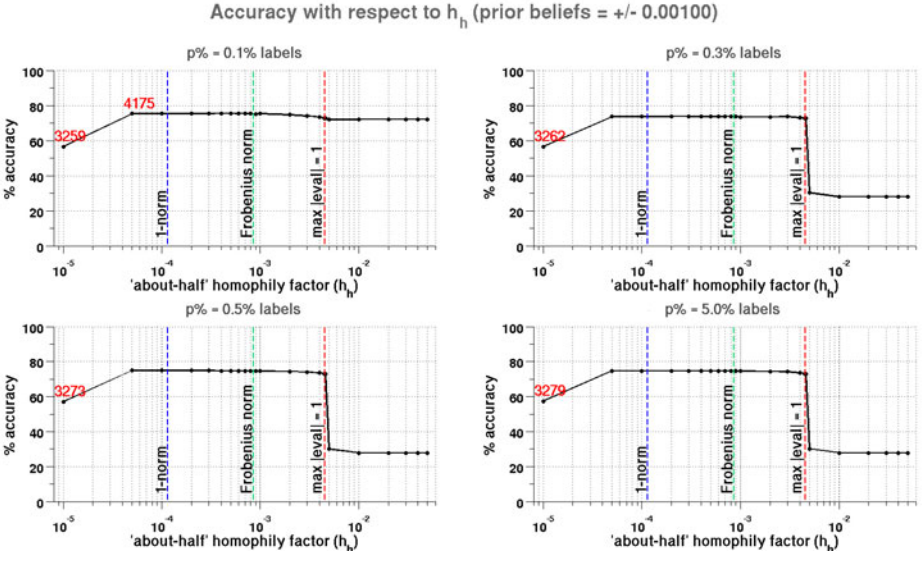


Fig. 3. FABP achieves maximum accuracy within the convergence bounds. The annotated red numbers correspond to the classified nodes when not all nodes were classified by FABP.

Observation 2. *Our convergence bounds consistently coincide with high-accuracy regions. Thus, we recommend choosing the homophily factor based on the Frobenius norm using* (8).

6.3 Q3: Sensitivity to Parameters

Figure 3 shows that FABP is insensitive to the "about-half" homophily factor, h_h, as long as the latter is within the convergence bounds. Moreover, in Fig. 4 we observe that the accuracy score is insensitive to the *magnitude* of the prior beliefs. For brevity, we show only the cases $p \in \{0.1\%, 0.3\%, 0.5\%\}$, as for all values except for $p = 5.0\%$, the accuracy is practically identical. Similar results were found for different "about-half" homophily factors, but the plots are omitted due to lack of space.

Observation 3. *The accuracy results are insensitive to the* magnitude *of the prior beliefs and the homophily factor - as far as the latter is within the convergence bounds we gave in Section 4.*

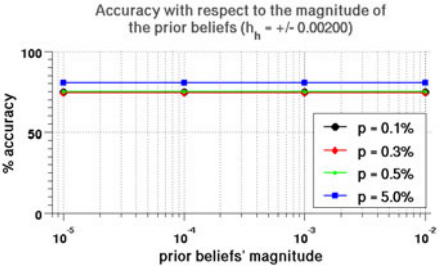


Fig. 4. Insensitivity of FABP to the *magnitude* of the prior beliefs

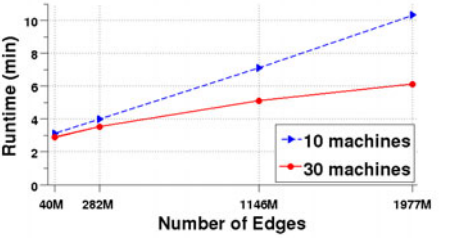
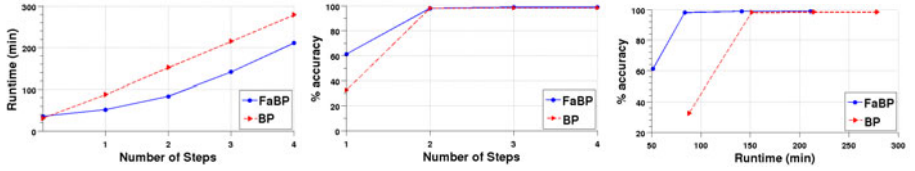


Fig. 5. FABP runtime vs # edges of Kronecker graphs for 10 and 30 machines on HADOOP

6.4 Q4: Scalability

To show the scalability of FABP we implemented FABP on HADOOP, an open source MAPREDUCE framework, which has been successfully used for large scale graph analysis [14]. We first show the scalability of FABP on the number of edges of Kronecker graphs. As seen in Fig. 5, FABP scales linearly on the number of edges. Next, we compare HADOOP implementation of FABP and BP [13] in terms of running time and accuracy on YahooWeb graph. Figures 6(a-c) show that FABP achieves the maximum accuracy level after two iterations of the Power Method and is $\sim 2\times$ faster than BP. This is explained as follows: BP needs to store the updated messages for 2 states on disks for large graphs, and thus, it stores $2|E|$ records in total, where $|E|$ is the number of edges. In contrast, FABP stores n records per iteration, where n is the number of nodes. Given that $n < 2|E|$, FABP is faster than BP.

Observation 4. FABP *is linear on the number of edges, with* $\sim 2\times$ *faster running time than* BP *on* HADOOP.



(a) Runtime vs # of iterations (b) Accuracy vs # iterations (c) Accuracy vs runtime

Fig. 6. Performance on the YahooWeb graph (best viewed in color): FaBP wins on speed and wins/ties on accuracy. In (c), each of the method has 4 points that correspond to one step from 1 to 4. FaBP achieves the maximum accuracy after 84 minutes, while BP achieves the same accuracy after 151 minutes.

7 Conclusions

Which of the many *guilt-by-association* methods one should use? We answered this question, and we developed FaBP, a new, fast algorithm to do such computations. The contributions of our work are the following:

- *Theory & Correspondences*: We showed that successful, major *guilt-by-association* approaches (RWR, SSL, and BP variants) are closely related, and we proved that some are even equivalent under certain conditions (Theorem 1, Lemmas 1, 2, and 3).
- *Algorithms & Convergence*: Thanks to our analysis, we designed FaBP, a fast and accurate approximation to the standard belief propagation (BP), which has convergence guarantee (Lemmas 5 and 6).
- *Implementation & Experiments*: We showed that FaBP is significantly faster, about 2×, and has the same or better accuracy (AUC) than BP. Moreover, we showed how to parallelize it with MapReduce (Hadoop), operating on *billion-node* graphs.

Thanks to our analysis, our guide to practitioners is the following: among all 3 *guilt-by-association* methods, we recommend belief propagation, for two reasons: (1) it has solid, Bayesian underpinnings and (2) it can naturally handle heterophily, as well as multiple class-labels. With respect to parameter setting, we recommend to choose homophily score, h_h, according to the Frobenius bound in (8).

Future work could focus on time-evolving graphs, and label-tracking over time. For instance, in a call-graph, we would like to spot nodes that change behavior, e.g. from "telemarketer" type to "normal user" type.

Acknowledgments. This work is partially supported by an IBM Faculty Award, by the National Science Foundation under Grants No. CNS-0721736 IIS0970179, under the project No. NSC 98-2221-E-011-105, NSC 99-2218-E-011-019, under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. DE-AC52-07NA27344, and by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, the U.S. Government, NSF, or any other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] Hadoop information, http://hadoop.apache.org/
[2] Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks 30(1-7) (1998)
[3] Chau, D.H., Nachenberg, C., Wilhelm, J., Wright, A., Faloutsos, C.: Polonium: Tera-scale graph mining and inference for malware detection. In: SDM (2011)
[4] Chechetka, A., Guestrin, C.: Focused belief propagation for query-specific inference. In: International Conference on Artificial Intelligence and Statistics (AISTATS) (May 2010)
[5] Christakis, N.A., Fowler, J.H.: The spread of obesity in a large social network over 32 years. New England Journal of Medicine 357(4), 370–379 (2007)
[6] Felzenszwalb, P., Huttenlocher, D.: Efficient belief propagation for early vision. International Journal of Computer Vision 70(1), 41–54 (2006)
[7] Fowler, J.H., Christakis, N.A.: Dynamic spread of happiness in a large social network: longitudinal analysis over 20 years in the Framingham Heart Study. BMJ (2008)
[8] Gao, J., Liang, F., Fan, W., Sun, Y., Han, J.: Graph-based Consensus Maximization among Multiple Supervised and Unsupervised Models. In: NIPS (2009)
[9] Gonzalez, J., Low, Y., Guestrin, C.: Residual splash for optimally parallelizing belief propagation. In: AISTAT (2009)
[10] Haveliwala, T.H.: Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. IEEE Transactions on Knowledge and Data Engineering, 784–796 (2003)
[11] Haveliwala, T., Kamvar, S., Jeh, G.: An analytical comparison of approaches to personalizing pagerank. Technical report, Stanford University (2003)
[12] Ji, M., Sun, Y., Danilevsky, M., Han, J., Gao, J.: Graph regularized transductive classification on heterogeneous information networks. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6321, pp. 570–586. Springer, Heidelberg (2010)
[13] Kang, U., Chau, D.H., Faloutsos, C.: Mining large graphs: Algorithms, inference, and discoveries. In: ICDE, pp. 243–254 (2011)
[14] Kang, U., Tsourakakis, C., Faloutsos, C.: Pegasus: A peta-scale graph mining system - implementation and observations. In: IEEE International Conference on Data Mining (2009)
[15] Koren, Y., North, S.C., Volinsky, C.: Measuring and extracting proximity in networks. In: KDD, pp. 245–255. ACM, New York (2006)
[16] Kschischang, F., Frey, B., Loeliger, H.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 47(2), 498–519 (2001)
[17] Leskovec, J., Chakrabarti, D., Kleinberg, J.M., Faloutsos, C.: Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 133–145. Springer, Heidelberg (2005)

[18] Malioutov, D.M., Johnson, J.K., Willsky, A.S.: Walk-sums and belief propagation in gaussian graphical models. Journal of Machine Learning Research 7, 2031–2064 (2006)
[19] McGlohon, M., Bay, S., Anderle, M.G., Steier, D.M., Faloutsos, C.: Snare: a link analytic system for graph labeling and risk detection. In: KDD (2009)
[20] Minkov, E., Cohen, W.: Learning to rank typed graph walks: Local and global approaches. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis, pp. 1–8. ACM, New York (2007)
[21] Pan, J., Yang, H., Faloutsos, C., Duygulu, P.: Gcap: Graph-based automatic image captioning. In: MDDE (2004)
[22] Pandit, S., Chau, D., Wang, S., Faloutsos, C.: Netprobe: a fast and scalable system for fraud detection in online auction networks. In: WWW (2007)
[23] Pearl, J.: Reverend Bayes on inference engines: A distributed hierarchical approach. In: Proceedings of the AAAI National Conference on AI, pp. 133–136 (1982)
[24] Tong, H., Faloutsos, C., Pan, J.: Fast random walk with restart and its applications. In: Perner, P. (ed.) ICDM 2006. LNCS (LNAI), vol. 4065, Springer, Heidelberg (2006)
[25] Weiss, Y.: Correctness of local probability propagation in graphical models with loops. Neural computation 12(1), 1–41 (2000)
[26] Yedidia, J., Freeman, W., Weiss, Y.: Understanding belief propagation and its generalizations. Exploring Artificial Intelligence in the New Millennium 8, 236–239 (2003)
[27] Yedidia, J., Freeman, W., Weiss, Y.: Constructing free-energy approximations and generalized belief propagation algorithms. IEEE Transactions on Information Theory 51(7), 2282–2312 (2005)
[28] Zhu, X.: Semi-supervised learning literature survey (2006)

Appendix A: Preliminaries - Analysis of FaBP

In this appendix we present the lemmas that are needed to prove Theorem 1 (FaBP), which gives the linearized version of BP. We start with the original BP equations, and we derive the proof by:

- using the *odds ratio* $p_r = p/(1-p)$, instead of probabilities. The advantage is that we have only one value for each node, $p_r(i)$, instead of two, $p_+(i)$ and $p_-(i)$; also, the normalization factor is not needed. Moreover, working with the *odds ratios* results in the substitution of the propagation matrix entries by a scalar homophily factor.
- assuming that all the parameters are close to 1/2, using Maclaurin expansions to linearize the equations, and keeping only the first order terms. By doing so we avoid the sigmoid/non-linear equations of BP.

Traditional BP equations: In [26], Yedidia derives the following update formulas for the messages sent from node i to node j and the belief of each node that it is in state x_i

$$m_{ij}(x_j) = \sum_{x_i} \phi_i(x_i) \cdot \psi_{ij}(x_i, x_j) \cdot \prod_{n \in N(i) \setminus j} m_{ni}(x_i) \tag{9}$$

$$b_i(x_i) = \eta \cdot \phi_i(x_i) \cdot \prod_{j \in N(i)} m_{ij}(x_i) \tag{10}$$

where the message from node i to node j is computed based on all the messages sent by all its neighbors in the previous step except for the previous message sent from node j to node i. $N(i)$ denotes the neighbors of i and η is a normalization constant that guarantees that the beliefs sum to 1.

Table 5. Additional Symbols and Definitions

Symbols	Definitions
p	P(node in positive class) = P("+")
m	message
$< var >_r$	odds ratio = $\frac{<var>}{1-<var>}$, where $< var >= b, \phi, m, h$
$B(a, b)$	blending function of the variables a and $b = \frac{a \cdot b+1}{a+b}$.

Lemma 7. *Expressed as ratios, the BP equations become:*

$$m_r(i, j) \leftarrow B[h_r, b_{r,adjusted}(i, j)] \tag{11}$$

$$b_r(i) \leftarrow \phi_r(i) \cdot \prod_{j \in N(i)} m_r(j, i) \tag{12}$$

where $b_{r,adjusted}(i, j)$ is defined as $b_{r,adjusted}(i, j) = b_r(i)/m_r(j, i)$. The division by $m_r(j, i)$ subtracts the influence of node j when preparing the message $m_r(i, j)$.

Proof. The proof is straightforward. Notice that $1 - v_+(i) = v_-(i)$ for $v \in \{b, \phi, m\}$, eg., $b_-(i) = 1 - b_+(i) = \eta \cdot (1 - \phi_+(i)) \cdot \prod_{j \in N(i)}(1 - m_+(i, j))$. □

Lemma 8 (Approximations)**.** *Fundamental approximations for all the variables of interest, $\{m, b, \phi, h\}$, are:*

$$v_r = \frac{v}{1 - v} = \frac{1/2 + v_h}{1/2 - v_h} \approx 1 + 4v_h \tag{13}$$

$$B(a_r, b_r) \approx 1 + 8a_h b_h \tag{14}$$

where $B(a_r, b_r)$ is the blending function for any variables a_r, b_r.

Sketch of proof. Use the definition of "about-half" approximations, apply the appropriate Maclaurin series expansions and keep only the first order terms. □

Lemmas 9-11 are useful in order to derive the linear equation of FaBP. Note that we apply several approximations, but omit the "≈" symbol to make the proofs more readable.

Lemma 9. *The "about-half" version of the belief equation becomes, for small deviations from the half-point:*

$$b_h(i) \approx \phi_h(i) + \sum_{j \in N(i)} m_h(j, i)\,. \tag{15}$$

Proof. We use (12) and (13) and apply the appropriate Maclaurin series expansions:

$$b_r(i) = \phi_r(i) \prod_{j \in N(i)} m_r(j,i) \Rightarrow$$
$$log\,(1 + 4b_h(i)) = log\,(1 + 4\phi_h(i)) + \sum_{j \epsilon N(i)} log\,(1 + 4m_h(j,i)) \Rightarrow$$
$$b_h(i) = \phi_h(i) + \sum_{j \epsilon N(i)} m_h(j,i)\,. \qquad \square$$

Lemma 10. *The "about-half" version of the message equation becomes:*

$$m_h(i,j) \approx 2h_h[b_h(i) - m_h(j,i)]\,. \tag{16}$$

Proof. We combine (11), (13) and (14) to deduce

$$m_r(i,j) = B[h_r, b_{r,adjusted}(i,j)] \Rightarrow m_h(i,j) = 2h_h b_{h,adjusted}(i,j)\,. \tag{17}$$

In order to derive $b_{h,adjusted}(i,j)$ we use (13) and the approximation of the Maclaurin expansion $\frac{1}{1+\epsilon} \approx 1 - \epsilon$ for a small quantity ϵ:

$$b_{r,adjusted}(i,j) = b_r(i)/m_r(j,i) \Rightarrow$$
$$1 + b_{h,adjusted}(i,j) = (1 + 4b_h(i))(1 - 4m_h(j,i)) \Rightarrow$$
$$b_{h,adjusted}(i,j) = b_h(i) - m_h(j,i) - 4b_h(i)m_h(j,i)\,. \tag{18}$$

Substituting (18) to (17) and ignoring the terms of second order, leads to the about-half version of the message equation. $\square$

Lemma 11. *At steady state, the messages can be expressed in terms of the beliefs:*

$$m_h(i,j) \approx \frac{2h_h}{(1 - 4h_h^2)}\,[b_h(i) - 2h_h b_h(j)]\,. \tag{19}$$

Proof. We apply Lemma 10 both for $m_h(i,j)$ and $m_h(j,i)$ and we solve for $m_h(i,j)$. $\square$

Appendix B: Proofs of Section 3 (Theorems)

Here we give the proofs of the theorems and lemmas presented in Section 3.

Proof of Theorem 1. We substitute (16) to (15) and we obtain:

$$b_h(i) - \sum_{j \in N(i)} m_h(j,i) = \phi_h(i) \Rightarrow$$
$$b_h(i) + \sum_{j \in N(i)} \frac{4h_h^2 b_h(j)}{1 - 4h_h^2} - \sum_{j \in N(i)} \frac{2h_h}{1 - 4h_h^2} b_h(i) = \phi_h(i) \Rightarrow$$
$$(\mathbf{I} + a\mathbf{D} - c'\mathbf{A})\mathbf{b_h} = \boldsymbol{\phi_h}\,. \qquad \square$$

Proof of Lemma 2. Given l labeled points (x_i, y_i), $i = 1, \ldots, l$, and u unlabeled points $x_{l+1}, \ldots, x_{l+u}$ for a semi-supervised learning problem, based on an energy minimization formulation, we find the labels x_i by minimizing the following function E

$$E(\boldsymbol{x}) = \alpha \sum_{j \in N(i)} a_{ij}(x_i - x_j)^2 + \sum_{1 \le i \le l} (y_i - x_i)^2 \,, \tag{20}$$

where α is related to the coupling strength (homophily) of neighboring nodes, and $N(i)$ denotes the neighbors of i. If *all* points are labeled, (20) becomes, in matrix form,

$$\begin{aligned} E(\boldsymbol{x}) &= \boldsymbol{x}^T[\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]\boldsymbol{x} - 2\boldsymbol{x} \cdot \boldsymbol{y} + K(\boldsymbol{y}) \\ &= (\boldsymbol{x} - \boldsymbol{x}^*)^T[\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})](\boldsymbol{x} - \boldsymbol{x}^*) + K'(\boldsymbol{y}) \,, \end{aligned}$$

where $\boldsymbol{x}^* = [\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]^{-1}\boldsymbol{y}$, and K, K' are some constant terms which depend only on $\boldsymbol{y}$. Clearly, E achieves the minimum when

$$\boldsymbol{x} = \boldsymbol{x}^* = [\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]^{-1}\boldsymbol{y} \,.$$

The equivalence of SSL and Gaussian BP can be found in [25]. □

Proof of Lemma 3. Based on (2) and (3), the two methods will give identical results if

$$\begin{gathered} (1-c)[\mathbf{I} - c\mathbf{D}^{-1}\mathbf{A}]^{-1} = [\mathbf{I} + \alpha(\mathbf{D} - \mathbf{A})]^{-1} \Leftrightarrow \\ \left(\frac{c}{1-c}\right)[\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}] = \alpha(\mathbf{D} - \mathbf{A}) \Leftrightarrow \\ \left(\frac{c}{1-c}\right)\mathbf{D}^{-1} = \alpha\mathbf{I} \,. \end{gathered}$$

This cannot hold in general, unless the graph is "regular": $d_i = d$ $(i = 1, \ldots, n)$, or $\mathbf{D} = d \cdot \mathbf{I}$, in which case the condition becomes

$$\alpha = \frac{c}{(1-c)d} \Rightarrow c = \frac{\alpha d}{1 + \alpha d} \tag{21}$$

where d is the common degree of all the nodes. □

Appendix C: Proofs of Section 4 (Convergence)

Proof of Lemma 5. In order for the power series to converge, a sub-multiplicative norm of matrix $\mathbf{W} = c\mathbf{A} - a\mathbf{D}$ should be smaller than 1. In this analysis we use the 1-norm (or equivalently the ∞-norm). The elements of matrix $\mathbf{W}$ are either $c = \frac{2h_h}{1-4h_h^2}$ or $-ad_{ii} = \frac{-4h_h^2 d_{ii}}{1-4h_h^2}$. Thus, we require

$$\begin{gathered} \max_j (\sum_{i=1}^{n} |\mathbf{W}_{ij}|) < 1 \Rightarrow (c + a) \cdot \max_j d_{jj} < 1 \Rightarrow \\ \frac{2h}{1-2h} \max_j d_{jj} < 1 \Rightarrow h_h < \frac{1}{2(1 + \max_j d_{jj})}. \end{gathered}$$

□

Online Clustering of High-Dimensional Trajectories under Concept Drift

Georg Krempl[1], Zaigham Faraz Siddiqui[2], and Myra Spiliopoulou[2]

[1] University of Graz
georg.krempl@uni-graz.at
[2] University of Magdeburg
{myra,siddiqui}@iti.cs.uni-magdeburg.de

Abstract. Historical transaction data are collected in many applications, e.g., patient histories recorded by physicians and customer transactions collected by companies. An important question is the learning of models upon *the primary objects* (patients, customers) rather than the transactions, especially when these models are subjected to drift.

We address this problem by combining advances of online clustering on multivariate data with the trajectory mining paradigm. We model the measurements of each individual primary object (e.g. its transactions), taken at irregular time intervals, as a trajectory in a high-dimensional feature space. Then, we cluster individuals with similar trajectories to identify sub-populations that evolve similarly, e.g. groups of customers that evolve similarly or groups of employees that have similar careers.

We assume that the multivariate trajectories are generated by drifting Gaussian Mixture Models. We study (i) an EM-based approach that clusters these trajectories incrementally as a reference method that has access to all the data for learning, and propose (ii) an online algorithm based on a Kalman filter that efficiently tracks the trajectories of Gaussian clusters. We show that while both methods approximate the reference well, the algorithm based on a Kalman filter is faster by one order of magnitude compared to the EM-based approach.

Keywords: On-Line clustering, incremental clustering, high-dimensional trajectories, clusters of trajectories, high-dimensional clustering, stream clustering.

1 Introduction

Finding clusters on a mixture of multivariate distributions is a well-investigated problem. A typical assumption is that the data can be represented as vectors. Although this requirement seems fairly easy to fulfil, it is counter intuitive for many applications. For example, consider tasks like learning the progress of some chronicle disease or the seasonal purchase habits of customers. The data at hand are measurements of fixed dimensionality for each individual, e.g. transactions performed by customers or recordings of a patient's blood pressure. However,

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 261–276, 2011.

we cannot assume that there is the same number of measurements for each individual, nor that the measurements have been performed at the same intervals for them. Hence, vector representation becomes too restrictive. In this study, we propose to model the measurements of individuals over time as ongoing *trajectories* in a high-dimensional space, and learn patterns over the evolving data by clustering together individuals whose trajectories deploy similarly.

Gaffney, Smyth and colleagues have proposed probabilistic methods for the clustering of trajectories of measurements [6,2,3]. However, they assume that the trajectories are available in their entirety, hence the algorithm can exploit all the data to learn the model. There are two problems with this approach. First, there are many cases where model learning is done *while* data are being recorded: not an ultimate model over all data is then of interest, but rather knowledge on how models change from one time point to the next. Second, a model over all data may smooth away concept drifts that are valuable to the observer. Some later works, like [7,14] address the first problem by adapting the model as time progresses and new data arrive. However, the problem of adapting to drift is treated in a rather rudimentary way, namely by considering only special aspects of drift (e.g. cluster split).

In this study, we propose an adaptive learning approach over the trajectories of individuals *as* the trajectories receive further measurements. We allow for measurements that are multi-variate data, and recorded at irregular intervals. We assume that these measurements are generated from a mixture of Gaussians, but allow for gradual and abrupt concept changes (drifts and shifts). For example, consider aeroplanes from Heathrow to Montreal and from Madrid to Montreal. They fly two clusters of routes, with the former being closer to Iceland than the other. After May 21, this clusters' route changes due to the volcanic eruption. While the cluster is still there, its aeroplanes move differently for a part of their route. This is an example of "abrupt concept change". For "gradual concept change", consider a cluster A of elderly women and B of women in their late forties. Members of A show a more and more intensive preference for books in big font; this cluster is drifting as a whole, indicating that its members become older. Further, some members of cluster B show an increasing preference for sports and gradually build a separate cluster C, i.e. there is drift inside the cluster B.

We approach model adaptation by using an incremental, but offline EM-algorithm for clustering trajectories. We further propose a method that tracks the trajectories of the Gaussian clusters (as whole entities) using Kalman filter. This second method is fast and adapts well - it is appropriate for online processing. We compare this algorithm to the offline EM-algorithm as a reference algorithm which can exploit all data for model learning, and we show that the model quality of our method is comparable to the reference, while the execution time is lower by one order of magnitude.

The paper is organised as follows. In the next section, we discuss related work. In section 3 we first discuss Gaussian mixture models and the Kalman filter and then describe our two adaptive methods. The experiments are reported in

section 4. The last section concludes our paper with a summary and a list of future research issues.

2 Related Work

Our work is on learning mixture models for the multi-variate data trajectories of evolving objects. Related work includes mixture models, object tracking methods, and methods that cluster objects which move in a conventional geometric space (e.g. cars, persons, vessels).

Mixture models: The work of Gaffney and Smyth [6] was one of the earliest to consider the problem of clustering trajectories without a vector-based representation. The approach implicitly assumes that the objects and their trajectories follow some basic model, i.e. the Gaussian Mixture Model, and uses a variant of Expectation-Maximisation algorithm to cluster sets of trajectories. Our approach is inspired by that study. However, we do not limit our data to time series, as in [6], but allow for multivariate measurements that constitute a stream rather than a closed time series. Further, the number of measurements per individual may vary from one individual to the other.

The method of Han et al. [7] does trajectory clustering by using the mixture model for automatic speech recognition. They report on the general problem of different initial cluster assignments leading to different EM clusters. To sidestep this problem, their variant of EM increases the number of clusters incrementally. The algorithm starts by computing the best fitting polynomial for the complete data set and then successively splits the cluster with the largest weight of the mixture densities until K clusters are obtained.

The work of Xiong et al. deals with the problem of tracking objects from a video stream [14]. They model these objects using a Gaussian mixture model. For the first frame, the parameters are initialised using the traditional mixture model. For each subsequent frame these parameters are predicted with the help of Kalman filter. Unlike the method of Han et al. [7], which utilises splitting during initialisation, Xiong et al. integrate "split", "merge" and "delete" operation into their dynamic Gaussian mixture model.

The follow-up work of Cadez, Gaffney and Smyth [2] deals with the problem of clustering individuals that are associated with non-vector, non-multi-variate data measurements. The example of such data can be patients at a hospital which are associated with multiple but varying number of time-series. The proposed solution is based on generative mixture models and can accept the data in its native form, i.e. with varying data sizes and types across the individuals.

Object Tracking: Kalman filtering [8] is a widely used technique in signal analysis, computational visualistics and other domains. It is an iterative state estimation filter that supports the estimation of current and subsequent states, when the exact nature of the modelled system is undetermined. The power of a Kalman filter comes from its ability to estimate the subsequent states in the presence of noise. However, calibrating the parameters such as process and measurement noise covariance is not trivial and even a simple predictions model can outperform Kalman filter in the absences of an accurate model [5].

Initially the filter was designed for spacecraft navigation but has been used widely for multi-object tracking [9,12,1]. The method discussed in Pathan et al. [12] tracks vehicles whose paths overlap or get merged. They denote it as a confusion state and make use of a Kalman filter to identify the trajectories of the individual objects. Mederios et al. [11] presented a comprehensive approach on tracking objects, using a sensor network of cameras. A Kalman filter is used for aggregating the data distributively, which is then shared among sensors to propagate the state of the objects as they move. As stated earlier, Xiong et al. [14] keep track of their elliptical objects represented by Gaussian clusters through a Kalman filter. In addition, they use their filter in order to predict the parameters of the clusters in the next timepoint.

The work of Ellis et al. [4] uses Gaussian process regression to model the trajectories of pedestrian trajectory patterns. They couple the Gaussian process with Monte Carlo methods and a Kalman filter to achieve a long term prediction. They show in their empirical evaluation that their Monte Carlo method has a much better performance than their Kalman filter over long term predictions. The poor performance of their Kalman filter is predictable as it needs measurements for calculating the error and updating the estimates. However, for very short term prediction, Kalman filter shows competitive performance.

Clustering moving objects: Our approach is based on the trajectory clustering paradigm introduced by Gaffney and Smyth [6] and has very similar problem definition with that of Cadez et al. [2]. Our first algorithm, which is used for initial cluster identification, is based on the EM algorithm by Gaffney and Smyth [6]. However, we extended the original univariate algorithm to handle multi-variate data. Unlike the method of Cadez et al. [2], where individuals are associated with time-series only, in our setting individuals are associated with multi-dimensional objects. Individuals can also vary in the number of their associated observations (also denoted objects). The EM algorithm of [6] has the drawback of not being online. This is due the fact that all historic observations of an individual are required in re-fitting a mixture regression model. This might cause high computational costs and might be susceptible to sudden shift, i.e. sudden, non-smooth changes in the position of cluster centres. Therefore, we introduce an online cluster tracking algorithm for the trajectory clustering paradigm. This algorithm is based on Kalman filter technique and can be easily initialised by using the parameter estimates obtained by the EM algorithm.

Related to our work is the method of Xiong et al. [14]. However, there are some important differences. First, their method is not online and performs adjustment of Gaussian mixture models at each frame. The input to their methods are the individual points and the target is to find the trajectories, whereas in our method data points are already associated to a trajectory and the task is 1) to cluster them and 2) to estimate the path of the true cluster centre over time. Moreover, its not entirely clear what they use as input to their algorithm, i.e. whether they use individual points or a batch of points.

Table 1. List of Symbols

	Symbol	Description
General	D	Dimensionality of feature space (e.g. $D = 2$ in the bivariate case)
	P	Order of the polynomials in the underlying mixture regression problem
	K	Number of clusters in the mixture
	N	Total number of observations (over all individuals)
	n	Number of observations for a given individual
	M	Total number of individuals
EM-Algorithm	z	D-dimensional feature vector (measurement)
	s, t_s	Discrete time step s at continuous time t_s
	Θ,θ_k	Parameter set (of the complete model or of cluster k)
	α_k, x_k, Σ_k	Mixing proportion, mean, and covariance of cluster k
	β_k	Regression coefficients (including cluster means x_k) for cluster k
	$\beta_{k\,dp}$	p-th regression coefficient for feature d in cluster k
Kalman Filter	x_s, z_s	True state and measurement at (discrete) time step s
	A	State transition matrix
	w_s, Q	Process noise at time step s, process noise covariance matrix
	H	Measurement (or state-to-signal) matrix
	v_s, R	Measurement noise at time step s, measurement noise cov. matrix
	K	Kalman gain matrix
	$P_{\bar{s}}$, P_s	Estimate covariances (a priori, and posterior)

3 Method TRACER

The main contribution of this paper is the extension of the trajectory clustering method provided by Gaffney and Smyth [6] to multivariate data streams which require online algorithms. We suggest the following approach: First, an extended variant of the EM algorithm of [6] is used to perform an initial clustering. As a result of this algorithm, parameter estimates for the underlying mixture regression are available. These estimates are then used to initialise a Kalman filter for tracking the clusters and updating the model parameter estimates regularly.

In the subsequent sections, we first give a brief overview of the Gaussian mixture models and the Kalman filter for completeness before outlining our tracking algorithm. A list of parameters and symbols is given in Table 1.

3.1 Gaussian Mixture Model

The objectives of our tracking algorithm are first to track clusters of individuals over time. Second, the parameters estimates of the assumed underlying data generating process should be updated. The input to clustering in this context are sets of individuals that are observed over a longer time span, e.g., patients and customers. Each individual is associated with a multiple of measurements (or observations). The measurements which have been recorded for an individual result in a trajectory. The number of measurements per individual as well as the observation time, i.e. the time at which a measurement is recorded, can differ between the individuals. For example, customers could make a varying number of transactions and not all customers will make the transactions at the same time. Furthermore it is reasonable to assume that the population of individuals is not homogeneous, but consists of sub-populations (clusters). Each sub-population might evolve differently over time. When a sub-population is affected by drift,

the behaviour of its individuals might also change. This results in a mixture regression model. If the distribution of measurements of a sub-population at a given moment in time is assumed to follow a Gaussian distribution, the following model of a mixture of Gaussians is obtained:

Algorithm 1: Incremental Clustering Of Trajectories using EM Algorithm

Input : new batch of data Z_i, old model $\mathcal{C}_{i-1}$ old estimates Θ_{i-1}
Output: new model $\mathcal{C}_i$, new estimates Θ_i
1 Computer the membership probabilities for the individuals in Z_i using $\mathcal{C}_{i-1}$.
2 Update the likelihood estimates
3 Repeat until re-convergence for new estimates and new clustering.

Let $z_i = z_{i1}, z_{i2}, \cdots, z_{in}$ be the n observations of the i-th individual. These observations follow a Gaussians mixture model with cluster means $x_1, x_2, \cdots, x_K$ and probability density function:

$$p(z_i; \Theta) = \prod_{l=1}^{n} \sum_{k=1}^{K} \alpha_k p(z_{il}; \theta_k) \tag{1}$$

where α_k is the mixing proportion of the k^{th} cluster such that $\sum_{k=1}^{K} \alpha_k = 1$. The density $p(z; \Theta)$ of the k-th cluster in the mixture with mean x_k follows a normal probability distribution

$$p(z; \theta_k) = (2\pi)^{-\frac{D}{2}} |\Sigma_k^{-1}|^{\frac{1}{2}} exp\left\{ -\frac{1}{2}(z - x_k)^T \Sigma_k^{-1} (z - x_k) \right\} \tag{2}$$

where $\theta_k = x_k, \Sigma_k$ is the centroid and covariance matrix for k^{th} cluster at a given moment in time, respectively.

For a given set of M independent individuals $Z = z_1, z_2, \ldots, z_M$, the log-likelihood with respect to the Gaussian mixture is given by

$$logp(Z; \Theta) = log \prod_{i=1}^{M} p(z_i; \Theta) = \sum_{i=1}^{M} \sum_{l=1}^{n_i} log \sum_{k=1}^{K} \alpha_k p(x_{il}; \theta_k) \tag{3}$$

Θ is complete set of parameters needed to define the complete mixture model, i.e., $\Theta = \{\alpha_1, \ldots, \alpha_k, x_1, \ldots, x_k, \Sigma_1, \ldots, \Sigma_k\}$. However, the maximum likelihood estimate of Θ cannot be computed analytically. The task here is to determine the individual mixture components from the joint density, using the given set of individuals Z.

3.2 Expectation-Maximisation Algorithm

Let us assume the cluster means in the mixture model above follow a polynomial functions of order P. In this mixture regression problem, the objective is to estimate the regression coefficients $\beta_k \; \forall \quad k = 1, \cdots, K$ as well as the cluster membership probability for each individual. EM is the widely used technique

for computing the maximum likelihood estimates for this problem, where the cluster membership probabilities are hidden. In this particular case, where all measurements of an individual are dependent, the EM-variant of [6] can be used for an initial clustering.

3.3 Kalman Filter

An elaborate overview about Kalman filter is presented in [13], we describe it briefly for completeness. Kalman filter addresses the problem of trying to estimate the state $x \in \mathcal{R}^N$ of a discrete-time controlled process that is governed by the following linear-time difference equation:

$$x_s = Ax_{s-1} \tag{4}$$

with a measurement $z \in \mathcal{R}^D$ that is

$$z_s = Hx_s + v_s \tag{5}$$

where A represents the transition matrix, x_s is the state (i.e. the true cluster centre) at time step s. H is the measurement matrix, z_s is the observed measurement (of an individual belonging to this cluster). w_s and v_s are random variables that represent process and measurement noise, respectively. They are mutually independent, white and with normal probability distributions, i.e., $p(w) \sim N(0, Q)$ and $p(v) \sim N(0, R)$.

Kalman filter basically estimate the state vector by using system sensors and measurement data, which are corrupted by noise. This estimation is done by using a type of feedback control: first the filter estimates the state of the process at a given time step s and then obtains feedback in terms of measurements which are assumed to be noisy. Eq. 4 and 5 describe a linear model at time s. Since x_s cannot be measured directly, the information provided by z_s is used to update the x_s.

The a priori state estimate $\hat{x}_s^-$ and covariance error P_k^- are calculated using the following time update equations:

$$\hat{x}_s^- = A\hat{x}_{s-1} + w_{s-1} \tag{6}$$

$$P_s^- = AP_{s-1}A^T + Q \tag{7}$$

The measurement update equations provides the feedback and are responsible for adjusting the model based on the new measurement (i.e., the priori estimates are adjusted to obtain a better posterior estimate):

$$S = HP_s^-H^T + R \tag{8}$$

$$K_s = P_s^-H^TS^{-1} \tag{9}$$

$$\hat{x}_s = \hat{x}_s^- + K_s(z_s - H\hat{x}_s^-) \tag{10}$$

$$P_s = (I - K_sH)P_s^- \tag{11}$$

where I = Identity Matrix of similar dimensions as P_s^-, and K_s is the Kalman gain. Using the measurement, a posterior state estimate $\hat{x}_s$ as well as an error estimate P_s are computed. The time and measurement equations are executed iteratively.

3.4 Kalman Filter Initialisation

Before the Kalman filter outlined above can be used, its parameters defining the state transition model as well as the measurement model must be initialised. Using the clusters' β-coefficients of the mixture regression model learnt by the EM-algorithm of Gaffney and Smyth, the position of each cluster centre at time t can be estimated. The coordinates corresponding to this position in the D-dimensional feature space at time t can be written as a vector $f^{(0)}(t) = f_1^{(0)}(t), \cdots, f_D^{(0)}(t)$, where $f_d^{(0)}(t) = \beta_{d0} + t\beta_{d1} + \cdots + t^o\beta_{do}$.

The true state at this time t comprises these coordinates, but also higher order derivatives of $f^{(0)}(t)$ as *meta-features*, as for example speed and acceleration. Let $f^{(l)}(t)$ denote the l-th derivative of $f^{(0)}(t)$ with respect to t. Assuming that all derivatives of orders greater than o are zero, the true state f at the time t can then be written as:

$$f(t) = \Big(f_1^{(0)}(t), \quad f_2^{(0)}(t), \quad \cdots, \quad f_D^{(0)}(t), \quad f_1^{(1)}(t), \quad f_2^{(1)}(t), \quad \cdots, \quad f_D^{(1)}(t), \\ \cdots, \quad f_1^{(o)}(t), \quad f_2^{(o)}(t), \quad \cdots, \quad f_D^{(o)}(t) \quad \Big)^T$$

The initialisation steps for the Kalman state model are outlined in Function `InitKalman()`. The probability distribution for the initial state x_0 is modelled by a mean vector μ_0 and a covariance matrix Σ_0 (Line 1). The mean vector can be estimated by evaluating $f(t_0)$ at the first point in time t_0. As a covariance matrix an identity matrix is used.

Given the $(s-1)$-th state at time t_s, the successive state at time t_s is defined by the state transition process above to be $z_s = Az_{s-1} + w_{s-1}$. Given an initial state z_0, the state transition matrix A as well as the process noise covariance matrix Q are needed to estimate the successive states.

Let $\Delta = t_s - t_{s-1}$ be the sampling interval, corresponding to the time between two samples are drawn from successive states $s-1$ and s. Using the Taylor series up to o-th order for defining $\delta_s = \frac{\Delta^s}{s!}$, one can write the state-transition matrix $A = [a_{ij}]$, where

Function InitKalman

Input : model $\mathcal{C}_i$, estimates Θ_i

1 Infer initial state using GMM parameters (β). `/* Init of Kalman State Space */`
2 Determine state transition matrix A (given sampling interval Δ).
3 Compute the covariance for the process noise Q.
4 Compute state-sensor matrix H `/* Init of State-Sensor Model */`
5 Compute the covariance for the sensor noise R .

$$a_{i,j} = \begin{cases} \delta_s \; if \quad \exists \quad q \in N: \quad i - D * q - j = 0 \\ 0 \;\; \text{otherwise} \end{cases}$$

For example, the state transition matrix of a 2-dimensional feature space considering up to 3rd order movement is

$$A = \begin{pmatrix} a_0 & 0 & a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & a_0 & 0 & a_1 & 0 & a_2 & 0 & a_3 \\ 0 & 0 & a_0 & 0 & a_1 & 0 & a_2 & 0 \\ 0 & 0 & 0 & a_0 & 0 & a_1 & 0 & a_2 \\ 0 & 0 & 0 & 0 & a_0 & 0 & a_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_0 & 0 & a_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_0 \end{pmatrix}$$

with $a_0 = 1$, $a_1 = \Delta$, $a_2 = \frac{\Delta^2}{2}$, $a_3 = \frac{\Delta^3}{6}$.

Assuming a uniform process noise over the feature space, the covariance matrix Q for the process noise can be set equal to the identity matrix, multiplied by a process noise factor $\hat{q}$ (Function `InitKalman()`, Line 3). This parameter $\hat{q}$ has to be tuned using the training data.

As our interest is the update of the regression obtained by the EM-algorithm, we can safely assume no distortion between the true state and the measurement, allowing the measurement matrix H to be set equal to the identity matrix (`InitKalman()`, Line 4). However, there is measurement noise to be considered. In order to obtain the measurement noise covariance matrix R, the covariance in the training data between the state estimates using EM and the observed data points as measurement is calculated (`InitKalman()`, Line 5).

The (estimated) *true* states, including higher order derivatives, can be calculated recursively using the β-coefficients obtained by the EM algorithm, as shown above. However, for the *observed measurement*, only the position $\hat{f}^{(0)}(t)$ is directly available, whereas the derivatives of $\hat{f}^{(0)}(t)$ with respect to t have to be estimated. This is solely possible for individuals with more than one measurement in the training data, as we need $o+1$ measurements per individual to estimate the derivatives of order o. Given three measurements of the $(i-1)$-th derivative $\hat{f}^{(i-1)}$ at times t_{-1}, t_0 and t_1 , respectively, the i-th derivative at time t_0 can be approximated as $\hat{f}^{(i)}(t_0) = \frac{\hat{f}^{(i-1)}(t_2) - \hat{f}^{(i-1)}(t_{-1})}{t_1 - t_{-1}}$. If only two measurements exist, the estimated (i)-th derivative is identical for both points in time, thus it is assumed that the the $(i+1)$-th derivative is zero. The resulting estimation of the (i)-th derivative for the two measurements taken at time points t_0, t_1 is then $\hat{f}^{(i)}(t_0) = \hat{f}^{(i)}(t_1) = \frac{\hat{f}^{(i-1)}(t_1) - \hat{f}^{(i-1)}(t_0)}{t_1 - t_0}$.

Once the difference between the estimated true state and the observed measurements of an individual is calculated, it is used to update the state estimate of a cluster. Two principal approaches can be distinguished: First, the state estimates of all clusters can be updated by using a weight proportional to the cluster membership probability of the instance. In contrast to this *weighted*

Algorithm 2: Tracking Trajectories with Kalman Filter

```
Input : new batch of data Z_i, model C_i, estimates Θ_i
foreach z ∈ Z_i do
    if z belongs to a known individual then
        ẑ ← from the meta-features for z
        Update sensor noise covariance R
        Update state transitions A based on the time elapsed b/w last cluster update δ_t
        Get new estimates for the clusters using the Kalman Filter
    Compute cluster membership probability for z
    Update cluster membership probabilities for the individual.
```

update strategy, a *the-winner-takes-all* update strategy only updates the cluster with the highest cluster membership probability.

While the winner-takes-all strategy is straightforward, the weighted update strategy requires a definition on how weights are used in the Kalman equations. In equation 8 from above the measurement noise is considered by adding its covariance matrix R as an addend. The extent of R is inversely related to the confidence in measurements. By multiplying a factor c with R, the confidence in a measurement can be incorporated into this equation.

$$S = HP_s^- H^T + R * c \tag{12}$$

This factor c could be set to the inverse of the cluster membership probability p, i.e. $c = \frac{1}{p}$. However, the inverse of the squared cluster membership probability, i.e. $c = \frac{1}{p^2}$, can also be used. The motivation is as follows: Assume an observation z does not belong to the cluster which is currently under consideration for update. As the distance of z to the cluster centre increases, the cluster membership probability decreases. Using the inverse of the squared cluster membership probability decreases the weight of very distant points in an over-proportional way, possibly reducing the influence of outliers. However, the effect of the choice of an update strategy is studied further in the experimental evaluation below.

3.5 Update and Clustering

Given an estimated probability distribution of the previous state, the current state on obtaining a new measurement for a cluster can be calculated using equation 10, where for the calculation of the Kalman gain K equation 9 and for the calculation of S equation 12 from above are used.

The individual-to-cluster assignment and the calculation of the higher order derivatives as meta-features can be done as in the calculation of the measurement noise covariance matrix above. This leads to the Algorithm 2. As new data is presented to the algorithm it adapts the model accordingly.

Given a new measurement z of individual i at time t_s and state estimates for time t_{s-1}, the algorithm first determines whether (a) the individual i is *known*, i.e. a prior measurement exists for i or (b) individual i is previously unknown. Only measurements from known individuals are used as sensor input for model update, in order to increase robustness against outliers.

A model update is done by first calculating the difference between sensor input and estimated state position. The sensor input $\hat{z}$ is the combination of the observed position of z with the calculated meta features (Line 3), which can be calculated as described in subsection 3.4 above. The estimated state position is calculated as defined in (Lines 4-5). The new state estimates are then used to re-compute the cluster membership probabilities for the new measurement z (Line 7), by using eq. 1. Lastly, for each cluster the product of the cluster membership probabilities of all measurements of the individual is calculated to obtain the cluster membership probability of the individual (Line 8). For numeric stability reasons, the log of the cluster membership probability can be used. The cluster membership probability of an individual can be updated by a single multiplication (or addition, if its log is used) with the probability of the new measurement. In contrast to EM, each measurement is processed once, which results in an online-behaviour of this Kalman trajectory tracking algorithm.

If the individual i is *new*, i.e., no prior measurements exist, its cluster membership probability is equal to the membership probability of the measurement. This probability can be calculated as explained above (Lines 7-8). An individual is not used for a model update unless further measurements are acquired.

4 Experiments

For the experimental evaluation, the Kalman filter was compared in different settings to the EM algorithm by [6]. To be able to study our algorithms in a controlled environment first, synthetic datasets had to be used. Cluster purity was used as performance measure. In analogy to [10] [page 357, equation 16.1], we define this measure as $purity = \frac{1}{N}\sum_{j=1}^{K} \max_{i=1}^{K} C_{ij}$ Here C is the confusion matrix, and C_{ij} is the number of elements that are in the i-th true cluster which are assigned to the j-th cluster by the algorithm. N is the number of elements in total. This measure is normalised to values in the interval $[0; 1]$, where one is a perfect separation. This normalisation allows averaging over several data sets without scaling issues, which could result when a measure based on the distance between true and predicted cluster states would have been used. However, as purity increases with the number of clusters, the number of clusters between methods should be similar. As the EM algorithm is used in the initialisation of the Kalman filter, the same number of clusters in all solutions is guaranteed.

A Wilcoxon signed rank test was performed to test the statistical significance of differences in clustering quality.

4.1 Data Sets

The data generator[1] used in the creation of the data sets uses a mixture model with K components. For each component a multivariate Gaussian density function is used to generate observations. The component centres are themselves

[1] The data generator and algorithms (implemented in Octave) are available at: https://bitbucket.org/geos/tracer-trajectory-tracking/overview

functions of time. For the initial state, the position, speed and acceleration as well as possible higher derivatives are generated at random. Subsequent states are calculates using a state-transition-matrix as described above, and adding random state-transition noise. Furthermore, sudden shift occurs at a given point in time, offsetting the cluster centres by a random vector.

Observations are sampled along this drift path by first calculating the cluster mean at the point in time, and subsequent sampling of observations using the multivariate Gaussian. Each observation is then assigned to an individual of its generating cluster.

The parameters of this data generator are the dimensionality of feature space D, the order of the polynomials used P, the number of clusters K, the number of individuals M, the number of observations N, the extend of the state-transition noise e_w, the strength of the state-to-signal noise e_r and the strength of the sudden shift e_s.

For the experimental evaluation, fifty data sets with five different parameter settings were generated:

1. **Datasets** $A_1, \cdots, A_{10}$: $D = 1$, $K = 3$, $e_w = 25$
2. **Datasets** $B_1, \cdots, B_{10}$: $D = 1$, $K = 3$, $e_w = 5$
3. **Datasets** $C_1, \cdots, C_{10}$: $D = 1$, $K = 3$, $e_w = 1$
4. **Datasets** $D_1, \cdots, D_{10}$: $D = 2$, $K = 3$, $e_w = 5$
5. **Datasets** $E_1, \cdots, E_{10}$: $D = 2$, $K = 3$, $e_w = 1$

$N = 3000$ observations were generated for $M = 1500$ individuals. This results in a very modest expected number of 2 observations per individual.While the first 1000 observations were used for initial training, the subsequent second third of observations was used to evaluate the performance prior to sudden shift. After the twothousandth observation, sudden shift occurred. Thus the last third of observations was used to evaluate the performance in presence of sudden shift.

4.2 Methods

The Expectation Maximisation algorithm was implemented as described in [6]. As all observations of an individual should be taken into account, there is no straightforward extension to an online version of this algorithm, and a batch version was used instead. Therefore, the initial model was fitted on the first one thousand observations. This model was later used for initialising the Kalman filter. For validation, the EM algorithm itself was refitted on all twothousand observations, prior to determining the cluster of any observation in the second batch. Similarly, the model was refitted on all observations before the cluster assignments in the last batch were evaluated.

It should be noted that this behaviour gives some advantage over a Kalman filter, as from the information used in training this would correspond to a Kalman smoother. However, such a comparison is not in the scope of this paper, which aims at a fast online cluster tracking algorithm. Therefore, the Kalman filter described above was initialised on the initial EM model trained solely on the first

Table 2. Cluster purity prior to shift

Data Set	EM	Kalman Tracking					
		K-1	K-2	K-3	K-4	K-5	K-6
A	0.92^{*}	0.83	0.78^{**}	0.83	0.82	0.79	0.81^{**}
	$\pm$ 0.10	$\pm$ 0.11	$\pm$ 0.15	$\pm$ 0.11	$\pm$ 0.13	$\pm$ 0.13	$\pm$ 0.11
B	0.92^{**}	0.73	0.68^{**}	0.72	0.72	0.69^{*}	0.69^{*}
	$\pm$ 0.04	$\pm$ 0.10	$\pm$ 0.10	$\pm$ 0.10	$\pm$ 0.12	$\pm$ 0.12	$\pm$ 0.08
C	0.87^{**}	0.82	0.79	0.81^{*}	0.80	0.78^{*}	0.81^{**}
	$\pm$ 0.13	$\pm$ 0.12	$\pm$ 0.15	$\pm$ 0.12	$\pm$ 0.11	$\pm$ 0.10	$\pm$ 0.12
D	0.98^{**}	0.90	0.81^{*}	0.89	0.89	0.86	0.82^{**}
	$\pm$ 0.03	$\pm$ 0.13	$\pm$ 0.18	$\pm$ 0.13	$\pm$ 0.13	$\pm$ 0.14	$\pm$ 0.17
E	0.96^{**}	0.84	0.79^{**}	0.84	0.84	0.81	0.81^{*}
	$\pm$ 0.11	$\pm$ 0.16	$\pm$ 0.17	$\pm$ 0.17	$\pm$ 0.16	$\pm$ 0.16	$\pm$ 0.15

batch of observations. The subsequent twothousand observations were clustered one-by-one by the Kalman filter.

The performance of both algorithms on the two validation sets is shown in the tables below.

4.3 Results

The results of the experimental evaluation on the 50 datasets is shown in tables 2,3,4, and 5 below. Single stars * indicate significant ($p < 0.05$) differences compared to K-1, and double stars ** denote highly significant ($p < 0.01$) differences.

Table 3. Cluster purity after shift

Data Set	EM	Kalman Tracking					
		K-1	K-2	K-3	K-4	K-5	K-6
A	0.88	0.87	0.73^{**}	0.86	0.84	0.78^{**}	0.81^{**}
	$\pm$ 0.13	$\pm$ 0.09	$\pm$ 0.11	$\pm$ 0.10	$\pm$ 0.13	$\pm$ 0.13	$\pm$ 0.10
B	0.87	0.80	0.75^{**}	0.80	0.76^{*}	0.70^{**}	0.78
	$\pm$ 0.14	$\pm$ 0.13	$\pm$ 0.12	$\pm$ 0.12	$\pm$ 0.13	$\pm$ 0.12	$\pm$ 0.11
C	0.92	0.89	0.84^{*}	0.88	0.86^{*}	0.78^{**}	0.84^{*}
	$\pm$ 0.12	$\pm$ 0.15	$\pm$ 0.17	$\pm$ 0.14	$\pm$ 0.13	$\pm$ 0.11	$\pm$ 0.15
D	0.96	0.96	0.87	0.96	0.94^{*}	0.88^{**}	0.92^{**}
	$\pm$ 0.10	$\pm$ 0.08	$\pm$ 0.14	$\pm$ 0.08	$\pm$ 0.09	$\pm$ 0.12	$\pm$ 0.09
E	0.96^{**}	0.89	0.82^{*}	0.89	0.88	0.83^{*}	0.86^{*}
	$\pm$ 0.10	$\pm$ 0.14	$\pm$ 0.15	$\pm$ 0.14	$\pm$ 0.14	$\pm$ 0.15	$\pm$ 0.13

The experimental evaluation shows that both algorithms are capable of identifying and tracking the cluster structure correctly. Overall, the performance of the Expectation-Maximisation algorithm is better in terms of purity, with a purity of 0.984 compared to the Kalman filter (0.896) before shift, and 0.964 compared to 0.957 after the shift (overall p-value of $p = 0.035$).

Table 4. Algorithm speed prior to shift

Data Set	EM	Kalman Tracking					
		K-1	K-2	K-3	K-4	K-5	K-6
A	75.9**	6.8	6.4**	6.8	6.8	6.8	6.8*
	± 20.6	± 0.0	± 0.0	± 0.0	± 0.0	± 0.1	± 0.1
B	81.8**	6.8	6.5**	6.8	6.9*	6.8	6.9**
	± 34.2	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0
C	87.1**	7.3	7.0	7.1	6.8	6.8	7.2
	± 61.8	± 1.1	± 0.9	± 0.8	± 0.0	± 0.0	± 0.7
D	64.5**	4.1	3.9**	4.2	4.2	4.1	4.1
	± 26.4	± 0.0	± 0.0	± 0.1	± 0.2	± 0.0	± 0.0
E	54.4**	4.2	3.9**	4.2	4.1	4.1	4.2
	± 13.4	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0

Table 5. Algorithm speed after shift

Data Set	EM	Kalman Tracking					
		K-1	K-2	K-3	K-4	K-5	K-6
A	88.2**	7.0	6.5**	7.0**	6.9*	6.9*	7.0
	± 32.7	± 0.1	± 0.0	± 0.0	± 0.1	± 0.1	± 0.1
B	107.3**	7.0	6.5**	7.0	7.0	7.0	7.1*
	± 73.9	± 0.1	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0
C	126.2**	7.1	7.4	7.9	7.2	8.2	7.0**
	± 124.7	± 0.2	± 1.8	± 1.9	± 0.6	± 2.5	± 0.1
D	73.5**	4.2	4.0**	4.3**	4.2*	4.2*	4.2*
	± 23.6	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0
E	81.9**	4.3	4.0**	4.3*	4.3**	4.3**	4.3
	± 44.8	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0	± 0.0

However, the performance is significantly ($p < 0.001$) worse in terms of computational time, with 65.5 compared to 4.1 seconds for the first validation set and 73.5 compared to 4.3 seconds for the second validations set on a quad core system. One reason might be the offline-behaviour of the EM, which makes use of all 2000 (or 3000) observations for model fitting prior to assigning clusters to the new observations. While this increases the clustering quality, it also leads to ten to twenty times higher computational time compared to the Kalman filter.

The experiments have clearly shown that the weighted cluster update strategy performs best for the Kalman filter. The best strategy is to use squared cluster membership probabilities as weights (K-1). Otherwise observations with low cluster membership probabilities can still have a strong influence on the cluster centre, if they are sufficiently far away. Therefore, using the cluster membership probabilities directly as weights (K-3) leads to a small, but significant ($p = 0.04$) reduction in cluster purity (0.893 compared to 0.896 before and 0.9571 compared to 0.9572 after the shift, in average over all data sets). The hard update strategy using a the-winner-takes-it-all approach (K-2) has shown a highly significant

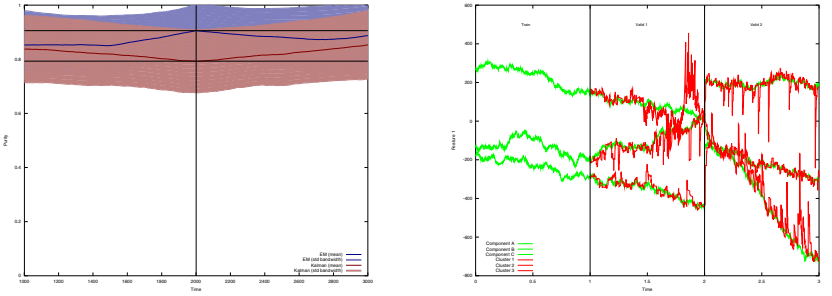


Fig. 1. (a) The development of purity over 30 data sets over time and (b) the distance between true and estimated states for the Kalman filter on one exemplary data set

($p < 0.001$) lower performance (0.808 and 0.869). Further comparisons included a multiplier different from one for the estimated state transition noise e_w (K-4 and K-5), which resulted in worse performance (all results are at least significant with $p < 0.05$). This indicates that the different extend of state transition noise e_w was estimated correctly by the algorithm, making further parameter tuning unnecessary. Finally the meta-features speed and acceleration, which are computed for the movement estimation, were included in the cluster membership probability estimation (K-6). However, this resulted in a highly significant ($p < 0.001$) performance decrease to 0.92 before and 0.82 after shift. In summary, one single parameter setting (K-1) was best on all data sets.

The effect of the sudden shift on the two algorithms is different: The Kalman filter corrects for this shift, and gains from the additional observations over time. This results in a highly significant increase in clustering quality (average purity increases from 0.82 to 0.88, $p < 0.0001$). The EM algorithm is effected differently: A shift constitutes a jump discontinuity in the path of the cluster centre over time, violating the assumed smoothness of this path. The EM algorithm tries to fit a polynomial regression function to the data, which will result in large errors around points of shift. In our experiments, this resulted in a (not significant) decrease in purity from 0.93 to 0.92. This is illustrated in figure 1, where the average purity over a moving window of 1000 observations is plotted over time for both EM and Kalman filter. The horizontal lines indicate cluster purity prior to shift, the vertical line corresponds to the moment shift occurs.

Finally it should be noted that in the experiments only a very modest number of observations per individual was chosen. The results show that both algorithms perform better as more observations per individual become available. Thus the performance in practise can be expected to improve further as the number of available observations increases over time.

5 Conclusions

We studied the problem of clustering trajectories over time when drifts and shifts occur. We proposed TRACER for trajectory cluster tracking in two variants:

an incremental EM algorithm for trajectory clustering and an online tracking algorithm that uses Kalman filter. Our experiments show that both variants of TRACER track clusters over time properly, while the online variant requires significantly less computational time.

Our first experiments have been performed on synthetic data, since we needed insights on the behavior of the algorithms in a controlled environment. We will now experiment with real data on the behavior of individuals (e.g. customers) over time. We further want to improve the robustness of the online TRACER against outliers and to study its behavior in the presence of multiple shifts.

References

1. Buzan, D., Sclaroff, S., Kollios, G.: Extraction and clustering of motion trajectories in video. In: Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, vol. 2, pp. 521–524 (2004)
2. Cadez, I.V., Gaffney, S., Smyth, P.: A general probabilistic framework for clustering individuals and objects. In: KDD 2000, pp. 140–149. ACM, New York (2000)
3. Chudova, D., Gaffney, S., Mjolsness, E., Smyth, P.: Translation-invariant mixture models for curve clustering. In: KDD 2003, pp. 79–88. ACM, New York (2003)
4. Ellis, D., Sommerlade, E., Reid, I.D.: Modelling pedestrian trajectory patterns with gaussian processes. In: VS 2009, pp. 1229–1234 (2009)
5. Funk, N.: A study of the kalman filter applied to visual tracking. Report (2003)
6. Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models. In: KDD 1999, pp. 63–72. ACM, New York (1999)
7. Han, Y., de Veth, J., Boves, L.: Trajectory clustering for automatic speech recognition (2005)
8. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Trans. of the ASME – Journal of Basic Engineering 82(series D), 35–45 (1960)
9. Li, X., Wang, K., Wang, W., Li, Y.: A multiple object tracking method using kalman filter. In: IEEE, ICIA 2010, pp. 1862–1866 (2010)
10. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
11. Medeiros, H., Park, J., Kak, A.: Distributed object tracking using a cluster-based kalman filter in wireless camera networks. IEEE TSP 2 (2008)
12. Pathan, S.S., Al-Hamadi, A., Michaelis, B.: OIF - an online inferential framework for multi-object tracking with kalman filter. In: Jiang, X., Petkov, N. (eds.) CAIP 2009. LNCS, vol. 5702, pp. 1087–1095. Springer, Heidelberg (2009)
13. Welch, G., Bishop, G.: An introduction to the kalman filter. Tech Report (1995)
14. Xiong, G., Feng, C., Ji, L.: Dynamical gaussian mixture model for tracking elliptical living objects. Pattern Recognition Letters 27, 838–842 (2006), doi:10.1016/j.patrec.2005.11.015

Gaussian Logic for Predictive Classification

Ondřej Kuželka, Andrea Szabóová, Matěj Holec, and Filip Železný

Faculty of Electrical Engineering, Czech Technical University in Prague
Technicka 2, 16627 Prague, Czech Republic
{kuzelon2,szaboand,holecmat,zelezny}@fel.cvut.cz

Abstract. We describe a statistical relational learning framework called Gaussian Logic capable to work efficiently with combinations of relational and numerical data. The framework assumes that, for a fixed relational structure, the numerical data can be modelled by a multivariate normal distribution. We demonstrate how the Gaussian Logic framework can be applied to predictive classification problems. In experiments, we first show an application of the framework for the prediction of DNA-binding propensity of proteins. Next, we show how the Gaussian Logic framework can be used to find motifs describing highly correlated gene groups in gene-expression data which are then used in a set-level-based classification method.

Keywords: Statistical Relational Learning, Proteomics, Gene Expression.

1 Introduction

Modelling of relational domains which contain substantial part of information in the form of real valued variables is an important problem with applications in areas as different as bioinformatics or finance. So far there have not been many relational learning systems introduced in the literature that would be able to model multi-relational domains with numerical data efficiently. One of the frameworks able to work in such domains are hybrid Markov logic networks [21]. However, there is currently no known approach to learning structure of hybrid Markov logic networks which is mainly due to their excessive complexity. In this paper we describe a relatively simple framework for learning in rich relational domains containing numerical data. The framework relies on multivariate normal distribution for which many problems have tractable or even analytical solutions. Our novel system exploits regularities in covariance matrices (i.e. regularities regarding correlations) for construction of models capable to deal with variable number of numerical random variables. We mainly show how this novel system can be applied in predictive classification. We show that it can be applied to classification directly (Bayesian learning) or indirectly (feature construction for gene-expression data).

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 277–292, 2011.

2 A Probabilistic Framework

In this section, we describe a probabilistic model which will constitute theoretical foundations for our framework. Let $n \in N$. If $\boldsymbol{v} \in R^n$ then v_i $(1 \leq i \leq n)$ denotes the i-th component of $\boldsymbol{v}$. If $I \subseteq [1; n]$ then $\boldsymbol{v}_I = (v_{i_1}, v_{i_2}, \ldots v_{i_{|I|}})$ where $i_j \in I$ $(1 \leq j \leq |I|)$. To describe training examples as well as learned models, we use a conventional first-order logic language $\mathcal{L}$ whose alphabet contains a distinguished set of constants $\{\mathsf{r}_1, \mathsf{r}_2, \ldots \mathsf{r}_n\}$ and variables $\{R_1, R_2, \ldots R_m\}$ $(n, m \in N)$. An *r-substitution* ϑ is any substitution as long as it maps variables (other than) R_i only to terms (other than) r_j. For the largest k such that $\{R_1/\mathsf{r}_{i_1}, R_2/\mathsf{r}_{i_2}, \ldots, R_k/\mathsf{r}_{i_k}\} \subseteq \vartheta$ we denote $I(\vartheta) = (i_1, i_2, \ldots i_k)$. A (Herbrand) *interpretation* is a set of ground atoms of $\mathcal{L}$. $I(H)$ $(I(\varphi))$ denotes the naturally ordered set of indexes of all constants r_i found in an interpretation H ($\mathcal{L}$-formula φ).

Our training examples have both *structure* and *real parameters*. An example may e.g. describe a measurement of the expression of several genes; here the structure would describe functional relations between the genes and the parameters would describe their measured expressions. The structure will be described by an interpretation, in which the constants r_i represent uninstantiated real parameters. The parameter values will be determined by a real vector. Formally, an example is a pair $(H, \boldsymbol{\theta})$ where H is an interpretation, $\boldsymbol{\theta} \in \Omega_H$, and $\Omega_H \subseteq R^{|I(H)|}$. The pair $(H, \boldsymbol{\theta})$ may also be viewed as a non-Herbrand interpretation of $\mathcal{L}$, which is the same as H except for including R in its domain and assigning θ_i to r_i.

Examples are assumed to be sampled from the distribution

$$P\left(H, \Omega_H\right) = \int_{\Omega_H} f_H\left(\boldsymbol{\theta}|H\right) P\left(H\right) \mathbf{d}\boldsymbol{\theta}$$

which we want to learn. Here, $P(H)$ is a discrete probability distribution on the countable set of finite Herbrand interpretations of $\mathcal{L}$. If $\mathcal{L}$ has functions other than constants, we assume that $P(H)$ is non-zero only for finite H. $f_H\left(\boldsymbol{\theta}|H\right)$ are the conditional densities of the parameter values. The advantage of this definition is that it cleanly splits the possible-world probability into the discrete part $P(H)$ which can be modeled by state-of-the-art approaches such as Markov Logic Networks (MLN's) [6], and the continuous conditional densities $f_H\left(\boldsymbol{\theta}|H\right)$ which we elaborate here. In particular, we assume that $f\left(\boldsymbol{\theta}|H\right) = N\left(\boldsymbol{\mu}_H, \boldsymbol{\Sigma}_H\right)$, i.e., $\boldsymbol{\theta}$ is normally distributed with mean vector $\boldsymbol{\mu}_H$ and covariance matrix $\boldsymbol{\Sigma}_H$. The indexes H emphasize the dependence of the two parameters on the particular Herbrand interpretation that is parameterized by $\boldsymbol{\theta}$.

To learn $P\left(H, \Omega_H\right)$ from a sample E, we first discuss a strategy that suggests itself readily. We could rely on existing methods (such as MLN's) to learn $P(H)$ from the multi-set $\mathcal{H}$ of interpretations H occurring in E. Then, to obtain $f\left(\boldsymbol{\theta}|H\right)$ for each $H \in \mathcal{H}$, we would estimate $\boldsymbol{\mu}_H, \boldsymbol{\Sigma}_H$ from the multi-set $\hat{\Omega}_H$ of parameter value vectors $\boldsymbol{\theta}$ associated with H in the training sample E. The problem of this approach is that, given a fixed size of the training sample, when $\mathcal{H}$ is large, the multi-sets $\hat{\Omega}_H, H \in \mathcal{H}$ will be small, and thus the estimates of $\boldsymbol{\mu}_H, \boldsymbol{\Sigma}_H$ will be poor. For example, $\mathcal{H}$ may describe hundreds of metabolic pathway structures

and each $\hat{\Omega}_H$ may contain a few vectors of expressions related to proteins acting in H, and measured through costly experiments. For this kind of problem we develop a solution here. We explore a method, in which parameters determining $P(H, \Omega_H)$ can be estimated using the entire training set. The type of $P(H, \Omega_H)$ is obviously not known; note that it is generally not a Gaussian mixture since the $\boldsymbol{\theta}$ in the normal densities $f_H(\boldsymbol{\theta}|H)$ have, in general, different dimensions for different H. However, our strategy is to learn *Gaussian features* of the training set. A Gaussian feature (*feature*, for short) is a $\mathcal{L}$-formula φ, which for each example $(H, \boldsymbol{\theta})$ extracts some components of $\boldsymbol{\theta}$ into a vector $\boldsymbol{u}(\varphi)$, such that $\boldsymbol{u}(\varphi)$ is approximately normally distributed across the training sample. For each feature φ, $\boldsymbol{\mu}_{\boldsymbol{u}(\varphi)}$ and $\boldsymbol{\Sigma}_{\boldsymbol{u}(\varphi)}$ are then estimated from the entire training sample. A set of such learned features φ can be thought of as a constraint-based model determining an approximation to $P(H, \Omega_H)$. We define *Gaussian features* more precisely in a moment after we introduce *sample sets*.

Given an example $e = (H, \boldsymbol{\theta})$ and a feature φ, the *sample set* of φ and e is the multi-set $\mathcal{S}(\varphi, e) = \{\boldsymbol{\theta}_{I(\vartheta)} | H \models \varphi\vartheta\}$ where ϑ are r-substitutions grounding all free variables[1] in φ, and $H \models \varphi\vartheta$ denotes that $\varphi\vartheta$ is true under H.

Now we can formally define *Gaussian features*. Let φ be a $\mathcal{L}$-formula, $\{e_i\}$ be a set of examples drawn independently from a given distribution and let $\boldsymbol{\theta_i}$ be vectors, each drawn randomly from $\mathcal{S}(\varphi, e_i)$. We say that φ is a *Gaussian feature* if $\boldsymbol{\theta_i}$ is multivariate-normally distributed[2].

Given a non-empty sample set $\mathcal{S}(\varphi, e)$, we define the *mean vector* as

$$\boldsymbol{\mu}(\varphi, e) = \frac{1}{|\mathcal{S}(\varphi, e)|} \sum_{\boldsymbol{\theta} \in \mathcal{S}(\varphi, e)} \boldsymbol{\theta} \tag{1}$$

and the $\boldsymbol{\Sigma}$-*matrix* as

$$\boldsymbol{\Sigma}(\varphi, e) = \frac{1}{|\mathcal{S}(\varphi, e)|} \sum_{\boldsymbol{\theta} \in \mathcal{S}(\varphi, e)} (\boldsymbol{\theta} - \boldsymbol{\mu}(\varphi, e)) (\boldsymbol{\theta} - \boldsymbol{\mu}(\varphi, e))^T \tag{2}$$

Finally, using the above, we define estimates over the entire training set $\{e_1, e_2, \ldots e_m\}$

$$\widehat{\boldsymbol{\mu}}_\varphi = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{\mu}(\varphi, e_i) \tag{3}$$

$$\widehat{\boldsymbol{\Sigma}}_\varphi = \frac{1}{m} \sum_{i=1}^{m} \left(\boldsymbol{\Sigma}(\varphi, e_i) + \boldsymbol{\mu}(\varphi, e_i) \boldsymbol{\mu}(\varphi, e_i)^T \right) - \widehat{\boldsymbol{\mu}}_\varphi \widehat{\boldsymbol{\mu}}_\varphi^T \tag{4}$$

Let us exemplify the concepts introduced so far using an example concerning modelling of gene regulatory networks. Let the only Herbrand interpretations

[1] Note that an interpretation H does not assign domain elements to variables in $\mathcal{L}$. The truth value of a *closed* formula (i.e., one where all variables are quantified) under H does not depend on variable assignment. For a general formula though, it does depend on the assignment to its free (unquantified) variables.

[2] Note that whether a $\mathcal{L}$-formula φ is a Gaussian feature depends also on the particular distribution of the examples.

H with non-zero $P(H)$ be those composed of literals of the form $g(G_i, R_i)$ and $expr(G_i, G_j)$ where $g(G_i, R_i)$ is a predicate intended to capture *expression level* R_i of a gene G_i and $expr(G_i, G_j)$ is used to indicate that two genes are in relation of *expression* (i.e. that the first gene G_i is a *transcription factor* of gene G_j). For example, the next example $e_1 = (H_1, \boldsymbol{\theta}_1)$ corresponds to a measurement on a sample set of genes containing three genes $H_1 = g(g_1, r_1)$, $g(g_2, r_2)$, $g(g_3, r_3)$, $expr(g_1, g_3)$, $expr(g_2, g_3)$, $\boldsymbol{\theta}_1 = (0, 1, 0)$. Let us further suppose that we have another example $e_2 = (H_2, \boldsymbol{\theta}_2)$ which corresponds to another set of genes: $H_2 = g(g_1, r_1)$, $g(g_2, r_2)$, $g(g_3, r_3)$, $g(g_4, r_4)$, $expr(g_1, g_2)$, $expr(g_2, g_3)$, $expr(g_3, g_4)$, $\boldsymbol{\theta}_2 = (1, 1, 0, 1)$.

Assume that the following two formulas have been identified as Gaussian features $\varphi_1 = g(G_1, R_1) \wedge g(G_2, R_2) \wedge expr(G_1, G_2)$, $\varphi_2 = g(G_1, R_1) \wedge g(G_2, R_2) \wedge \neg expr(G_1, G_2) \wedge G_1 \neq G_2$. Their sample sets for both examples are $\mathcal{S}(\varphi_1, e_1) = \{(0, 0), (1, 0)\}$, $\mathcal{S}(\varphi_2, e_1) = \{(0, 1), (1, 0)\}$, $\mathcal{S}(\varphi_1, e_2) = \{(1, 1), (1, 0), (0, 1)\}$ $\mathcal{S}(\varphi_2, e_2) = \{(1, 0), (0, 1), (1, 1), (1, 0), (1, 1), (1, 1), (0, 1), (1, 1), (1, 1)\}$. The first element of $\mathcal{S}(\varphi_1, e_1)$ is obtained with $\vartheta = \{G_1/g_1, G_2/g_3, R_1/r_1, R_2/r_3\}$. Clearly, $e_1 \models \varphi_1\vartheta$, and we have $\boldsymbol{\theta}_{I(\vartheta)} = (0, 1, 0)_{I(\vartheta)} = (0, 0)$ since $I(\vartheta) = (1, 3)$. For each sample set, we may then calculate the corresponding mean vector and $\boldsymbol{\Sigma}$-matrix according to Eq's 1 and 2 (e.g., $\boldsymbol{\mu}(\varphi_1, e_1) = (0.5, 0)^T$, and $\boldsymbol{\mu}(\varphi_1, e_2) = (2/3, 2/3)^T$). After that we can calculate the training-set-wide estimates for both features by Eq's 3 and 4. Then we can estimate multivariate normal distribution e.g. of a set of genes described by $H_3 = g(g_1, r_1)$, $g(g_2, r_2)$, $g(g_3, r_3)$, $expr(g_1, g_2)$, $expr(g_2, g_3)$, $expr(g_3, g_1)$ on the basis of features φ_1 and φ_2 which gives us the covariance matrix

$$\boldsymbol{\Sigma}_{H_3} = \begin{bmatrix} 1 & c_e & c_e \\ c_e & 1 & c_e \\ c_e & c_e & 1 \end{bmatrix}$$

where the parameter c_e corresponds to the correlation coefficient estimated in feature φ_1.

Importantly, using the training-set-wide estimates, we can derive estimates of parameters $\boldsymbol{\mu}_{H_n}$ and $\boldsymbol{\Sigma}_{H_n}$ of the densities $f_{H_n}(\boldsymbol{\theta}|H_n)$ for any relational structure H_n consisting of the two types of literals[3], even if H_n does not occur in the training set. Of course, validity of such estimates highly depends on the question whether the constructed features are truly *Gaussian*. Otherwise, a problem might occur that the estimated matrix would not be positive definite, however, first it almost did not happen in our experiments and second, if such a situation really occurs, one can replace the *defective* estimated covariance by a nearest positive definite matrix [10].

3 Parameter Estimation

In this section, we will be concerned with estimation of parameters $\boldsymbol{\mu}_\varphi$ and $\boldsymbol{\Sigma}_\varphi$. Although the estimators are straightforward modifications of ordinary estimators

[3] Note that one can use any number of different predicate symbols in this framework, not just two.

of means and covariances, their correctness does not follow immediately from the correctness of these conventional estimators because the samples contained in sample sets $\mathcal{S}(\varphi, e_i)$ may be dependent. Namely, we will show that, for a Gaussian feature φ,

$$\widehat{\boldsymbol{\mu}}_\varphi = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{\mu}(\varphi, e_i)$$

is a consistent and unbiased estimator of the true mean $\boldsymbol{\mu}_\varphi$ and that

$$\widehat{\boldsymbol{\Sigma}}_\varphi = \frac{1}{m}\sum_{i=1}^{m} \left(\boldsymbol{\Sigma}(\varphi, e_i) + \boldsymbol{\mu}(\varphi, e_i)\boldsymbol{\mu}(\varphi, e_i)^T\right) - \widehat{\boldsymbol{\mu}}_\varphi \widehat{\boldsymbol{\mu}}_\varphi^T$$

is a consistent asymptotically unbiased estimator of the true covariance matrix $\boldsymbol{\Sigma}_\varphi$. In order to show this, we will need the next lemma.

Lemma 1. *Let $\mathcal{E}$ be a countable set of estimators converging in mean to the true value such that for each $\widehat{E}^i \in \mathcal{E}$, $\widehat{E}^j \in \mathcal{E}$ it holds $\mathbf{E}\widehat{E}^i_m = \mathbf{E}\widehat{E}^j_m$ (where $\widehat{E}^j_m$ denotes the estimate of the estimator $\widehat{E}^j$ using m-samples). Let $\mathcal{E}_i \subseteq \mathcal{E}$ be finite sets. Then*

$$\lim_{m\to\infty} Pr\left(|E^* - \frac{1}{|\mathcal{E}_m|}\sum_{\widehat{E}_i \in \mathcal{E}_m} \widehat{E}^i_m| > \epsilon\right) = 0$$

where $\epsilon > 0$ and E^* is the true value (i.e. the combination of the estimators is consistent).

First, we will rewrite the formula for $\widehat{\boldsymbol{\mu}}_\varphi$ as an average of a (large) number of consistent estimators converging in mean and after that we will apply Lemma 1. Let us impose a random total ordering on the elements of the sample sets $\mathcal{S}(\varphi, e_i) = \{s_1, \ldots, s_{m_i}\}$ so that we could index the elements of these sets. Next, let us have

$$\mathcal{X} = \{1, 2, \ldots, |\mathcal{S}(F, e_1)|\} \times \{1, 2, \ldots, |\mathcal{S}(\varphi, e_2)|\} \times \cdots \times \{1, 2, \ldots, |\mathcal{S}(\varphi, e_m)|\}$$

Then the formula for $\widehat{\boldsymbol{\mu}}_\varphi$ can be rewritten as follows:

$$\widehat{\boldsymbol{\mu}}_\varphi = \frac{1}{|\mathcal{X}|}\sum_{(i_1, i_2, \ldots, i_m)\in\mathcal{X}} \frac{1}{m}\left(s_{1,i_1} + s_{2,i_2} + \cdots + s_{m,i_m}\right)$$

where $s_{j,k}$ is a k-th element of the sample set $\mathcal{S}(\varphi, e_j)$. Now, each $\frac{1}{m}\left(s_{1,i_1} + s_{2,i_2} + \cdots + s_{m,i_m}\right)$ is an unbiased consistent estimator converging in mean according to the definition of Gaussian features (it is the ordinary estimator of mean). Now, we may apply Lemma 1 and infer that $\widehat{\boldsymbol{\mu}}_\varphi$ also converges in probability to $\boldsymbol{\mu}_\varphi$. The unbiasedness of the estimator then follows from basic properties of expectation.

The argument demonstrating consistency and asymptotic unbiasedness of the covariance estimator goes along similar lines as the argument for the mean estimator. First, we rewrite the sum

$$\widehat{\boldsymbol{\Sigma}}_{\varphi} = \frac{1}{m}\sum_{i=1}^{m}\left(\boldsymbol{\Sigma}(\varphi, e_i) + \boldsymbol{\mu}(\varphi, e_i)\boldsymbol{\mu}(\varphi, e_i)^T\right) - \widehat{\boldsymbol{\mu}}_{\varphi}\widehat{\boldsymbol{\mu}}_{\varphi}^T =$$

$$= \frac{1}{m}\sum_{i=1}^{m}\frac{1}{|\mathcal{S}(\varphi, e_i)|}\sum_{\boldsymbol{\theta_j}\in\mathcal{S}(\varphi,e_i)}\left(\boldsymbol{\theta_j} - \widehat{\boldsymbol{\mu}}_{\varphi}\right)\left(\boldsymbol{\theta_j} - \widehat{\boldsymbol{\mu}}_{\varphi}\right)^T$$

as a sum of asymptotically unbiased consistent estimators as follows:

$$\widehat{\boldsymbol{\Sigma}}_{\varphi} = \frac{1}{|\mathcal{X}|}\sum_{(i_1,\ldots,i_m)\in\mathcal{X}}\frac{1}{m}\Big(\left(s_{1,i_1} - \widehat{\boldsymbol{\mu}}\right)\left(s_{1,i_1} - \widehat{\boldsymbol{\mu}}\right)^T + \ldots$$

$$\ldots + \left(s_{n,i_m} - \widehat{\boldsymbol{\mu}}_{\varphi}\right)\left(s_{m,i_m} - \widehat{\boldsymbol{\mu}}_{\varphi}\right)^T\Big)$$

where $s_{j,k}$ is a k-th element of the sample set $\mathcal{S}(\varphi, e_j)$. Now, each sum

$$\frac{1}{m}\Big(\left(s_{1,i_1} - \widehat{\boldsymbol{\mu}}\right)\left(s_{1,i_1} - \widehat{\boldsymbol{\mu}}\right)^T + \cdots + \left(s_{m,i_m} - \widehat{\boldsymbol{\mu}}\right)\left(s_{m,i_m} - \widehat{\boldsymbol{\mu}}_{\varphi}\right)^T\Big)$$

is an asymptotically unbiased and consistent estimator converging in mean and the average of these estimators is thus asymptotically unbiased and consistent (again by basic properties of expectation and by Lemma 1).

In general, the problem of estimating $\boldsymbol{\mu}(\varphi, e_i)$ and $\boldsymbol{\Sigma}(\varphi, e_i)$ are NP-hard problems (they subsume the well-known NP-complete problem of θ-subsumption). However, they are tractable for a class of features, *conjunctive tree-like features* for which we have devised efficient algorithms (see Appendix for details).

4 Structure Search

In this section we briefly describe methods for constructing a set of features that *give rise* to models capable to appropriately model a given set of examples. The methods that we describe are specialized for working with tree-like features because estimation of the parameters is tractable for them as we have mentioned in the previous section. The feature construction algorithm for tree-like features is based on the feature-construction algorithm from [16]. It shares most of the favourable properties of the original algorithm like detection of redundant features. The output of the feature construction algorithm is a (possibly quite large) set of features and their parameters so we need to select a subset of these features which would provide us with good models.

First, let us describe how a mean vector and a covariance matrix for a relational structure (i.e. a Herbrand interpretation) R is computed using a given set of features. We assume that we have a set of features $\varphi_i \in \mathcal{F}$ which have been identified as Gaussian, their respective parameters $\boldsymbol{\mu}_{\varphi_i}$ and $\boldsymbol{\Sigma}_{\varphi_i}$ and a relational structure H (Herbrand interpretation) for which we want to construct the model. Furthermore, we assume that each distinguished constant r_i contained in H *is covered* by some feature $\varphi \in \mathcal{F}$, i.e. that for each r_i there is a feature

$\varphi \in \mathcal{F}$ and a r-substitution ϑ such that r_i is contained in $\varphi\vartheta$. Then the covariance matrix $\mathbf{\Sigma}_H$ can be constructed as follows. For each $\varphi \in \mathcal{F}$ we compute the set Θ_φ of all substitutions ϑ such that $H \models \varphi\vartheta$. After that, for each feature φ (with parameters $\boldsymbol{\mu}_\varphi$ and $\mathbf{\Sigma}_\varphi$) and each r-substitution $\vartheta \in \Theta_\varphi$, we set the entries $(\boldsymbol{\mu}_H)_i = (\boldsymbol{\mu}_\varphi)_I$ $(\mathbf{\Sigma}_H)_{i,j} = (\mathbf{\Sigma}_\varphi)_{I,J}$ where $\{R_I/r_i, R_J/r_j\} \subseteq \vartheta$. If the features are perfectly Gaussian and if the parameters $\boldsymbol{\mu}_\varphi$ and $\mathbf{\Sigma}_\varphi$ are known accurately, there is no problem. However, in practice, we may encounter situations where two features will *suggest* different values for some entries (be it for the reason that the features are not perfectly Gaussian or that the parameters were estimated from small samples). In such situations we will use an average of the values *suggested* by different features.

Having explained how $\boldsymbol{\mu}_H$ and $\mathbf{\Sigma}_H$ are constructed using a set of Gaussian features, we can explain a simple procedure for construction of the Gaussian feature set. On the input, we get a set of examples $e_i = (H_i, \boldsymbol{\theta}_i)$. The procedure starts by constructing a large set of *non-redundant* features exhaustively on a subset of the training data. Then, in the second step, a subset of features is selected. This is done by a greedy search algorithm optimizing a score function of the models on a different subset of training data not used previously for feature construction and parameter estimation.

5 A Straightforward Predictive Classification Method

A straightforward application of the Gaussian-logic framework is Bayesian classification. We use the algorithms described in the previous sections of this paper to learn a Gaussian-logic model for positive examples and a Gaussian-logic model for negative examples and then we use these models to classify examples by comparing likelihood ratios of the two models with a threshold. In this section we describe a case study involving an important problem from biology - prediction of DNA-binding propensity of proteins. Proteins which possess the ability to bind to DNA play a vital role in the biological processing of genetic information like DNA transcription, replication, maintenance and the regulation of gene expression. Several computational approaches have been proposed for the prediction of DNA-binding function from protein structure. It has been shown that electrostatic properties of proteins such as total charge, dipole moment and quadrupole moment or properties of charged patches located on proteins' surfaces are good features for predictive classification (e.g. [1], [3], [18], [20]). Szilágyi and Skolnick [19] created a logistic regression classifier based on 10 features including electrostatic dipole moment, proportions of charged amino acids Arg, Lys and Asp, spatial asymmetries of Arg and five more features not related to charged amino-acids: proportion of Ala and Gly and spatial asymmetry of Gly, Asn and Ser.

Here, we use Gaussian logic to create a model for capturing distributions of positively charged amino acids in protein sequences. Clearly, the distinguishing electrostatic properties of DNA-binding proteins, which have been observed in 3D structures of proteins in the previous works, should exhibit themselves also in the amino-acid protein sequences (possibly, not in a straightforward

manner because the 3D structure is a result of complicated folding of a protein's sequence). We split each protein into consecutive non-overlapping *windows*, each containing l_w amino acids (possibly except for the last window which may contain less amino acids). For each window of a protein P we compute the value a_i^+/l_w where a_i^+ is the number of positively charged amino-acids in the window i. Then for each protein P we construct an example $e_P = (H_P, \boldsymbol{\theta}_P)$ where $\boldsymbol{\theta}_P = \left(a_1^+/l_w, a_2^+/l_w, \ldots, a_{n_P}^+/l_w\right)$ and $H_P = w(1, r_1)$, $next(1, 2)$, $\ldots$, $next(n_P - 1, n_P)$, $w(n_P, r_P)$. We constructed only one feature $F_{non} = w(A, R_1)$ for non-DNA-binding proteins since we do not expect this class of proteins to be very homogeneous. For DNA-binding proteins, we constructed a more complex model by selecting a set of features using a greedy search algorithm. The greedy search algorithm optimized classification error on training data. Classification was performed by comparing, for a tested protein, the likelihood-ratio of the two models (DNA-binding and non-DNA-binding) with a threshold selected on the training data. We estimated the accuracy of this method using 10-fold cross-validation (always learning parameters and structure of the models and selecting the threshold and window length l_w using only the data from training folds) on a dataset containing 138 DNA-binding proteins (PD138 [19]) and 110 non-DNA-binding proteins (NB110 [1]). The estimated accuracies (*Gaussian Logic*) are shown in Table 1. The method performs similarly well as the method of Szilagyi et al. [19] (in fact, it outperforms it slightly but the difference is rather negligible) but uses much less information. Next, we were interested in the question whether the machinery of Gaussian logic actually helped improve the predictive accuracy in our experiments or whether we could obtain the same or better results using only the very simple feature $F = w(A, R_1)$ also to model the DNA-binding proteins, thus ignoring any correlation between charges of different parts of a protein (*Baseline Gaussian Logic* in Table 1). Indeed, the machinery of Gaussian Logic appears to be helpful from these results.

Table 1. Accuracies estimated by 10-fold cross-validation on PD138/NB110

Method	**Accuracy** [%]
Szilágyi et al.	81.4
Baseline Gaussian logic	78.7
Gaussian logic	**81.9**

It is interesting how well the Gaussian-logic model performed considering the fact that it used so little information (it completely ignored types of positively charged amino acids and it also ignored negative amino acids). The model that we presented here can be easily extended, e.g. by adding secondary-structure information. The splitting into consecutive windows used here is rather artificial and it would be more natural to split the sequence into windows corresponding to secondary-structure units (helices, sheets, coils). The features could then distinguish between consecutive windows corresponding to different secondary-structure units.

6 Feature Construction for Predictive Classification

In this section we present another application of the Gaussian-logic framework for predictive classification. We show how to use it to search for novel definitions of gene sets with high discriminative ability. This is useful in set-level classification methods for prediction from gene-expression data [11]. Set-level methods are based on aggregating values of gene expressions contained in pre-defined gene sets and then using these aggregated values as features for classification. Here, we, first, describe the problem and available data and then we explain how we can construct meaningful novel gene sets using Gaussian Logic.

The datasets contain class-labeled gene-samples corresponding to measurements of activities of thousands of genes. Typically, the datasets contain only tens of measured samples. In addition to this *raw* measured data, we also have relational description of some biological pathways from publicly available database KEGG [14]. Each KEGG pathway is a description of some biological process (a metabolic reaction, a signalling process etc.). It contains a set of genes annotated by relational description which contains relations among genes such as *compound*, *phosphorylation*, *activation*, *expression*, *repression* etc. The relations do not necessarily refer to the processes involving the genes per se but they may refer to relations among the products of these genes. For example, the relation *phosphorylation* between two genes A, B is used to indicate that a protein coded by the gene A adds phosphate group(s) to a protein coded by the gene B.

We constructed examples $(H_S, \boldsymbol{\theta}_S)$ from the gene-expression samples and KEGG pathways as follows. For each gene g_i, we introduced a logical atom $g(g_i, r_i)$ to capture its expression level. Then we added all relations extracted from KEGG as logical atoms $relation(g_i,\ g_j,\ relationType)$. We also added a numerical indicator of class-label to each example as a logical atom $label(\pm 1)$ where $+1$ indicates a positive example and -1 a negative example. Finally, for each gene-expression sample S we constructed the vector of the gene-expression levels $\boldsymbol{\theta}_S$. Using the feature construction algorithm outlined in Section 4 we constructed a large set of tree-like features[4] involving exactly one atom $label(L)$, at least one atom $g(G_i, R_i)$ and relations *expression*, *repression*, *activation*, *inhibition*, *phosphorylation*, *dephosphorylation*, *state* and *binding/association*. After that we had to select a subset of these features. Clearly, the aggregated values of meaningful gene sets should correlate with the class-label. A very often used aggregation method in set-level classification methods is the *average*. Therefore what we need to do is to select features based on the correlation of the average expression of the genes assumed by the feature and the class-label but this is easy since we have the estimate of the features' covariance matrices $\boldsymbol{\Sigma}_\varphi$ and computing the average expression of the assumed genes is just an affine transform. It suffices to extract correlation from the covariance matrix given as $B\boldsymbol{\Sigma}_\varphi B^T$ where B is a matrix representing the *averaging*. The absolute values of correlations give us means to heuristically order the features. Based on this ordering

[4] We have used a subset of 50 pathways from KEGG to keep the memory consumption of the feature-construction algorithm under 1GB.

Table 2. Accuracies of set-level-based classifiers with Gaussian-logic features and FCF-based features, estimated by leave-one-out cross-validation

Dataset	**Gaussian logic**	**FCF**
Collitis [4]	80.0	89.4
Pleural Mesothieloma [9]	94.4	92.6
Parkinson 1 [17]	52.7	54.5
Parkinson 2 [17]	66.7	63.9
Parkinson 3 [17]	62.7	77.1
Pheochromocytoma [5]	64.0	56.0
Prostate cancer [2]	85.0	80.0
Squamus cell carcinoma [15]	95.5	88.6
Testicular seminoma [8]	58.3	61.1
Wins	5	4

we found a collection of gene sets given by the features (ignoring gene sets which contained only genes contained in a union of already constructed gene sets).

We have constructed the features using a gene-expression dataset from [7] which we did not use in the subsequent predictive classification experiments. A feature defining gene sets which exhibited one of the strongest correlations with the class-label was the following:

$$F = label(R_1) \wedge g(A, R_2) \wedge relation(A, B, phosphorylation) \wedge \\ g(B, R_3) \wedge relation(A, C, phosphorylation) \wedge g(C, R_4)$$

We have compared gene sets constructed by the outlined procedure with gene sets based on so called *fully-coupled fluxes (FCFs)* which are biologically-motivated gene sets used previously in the context of set-level classification [11]. We constructed the same number of gene sets for our features as was the number of FCFs. The accuracies of an SVM classifier (estimated by leave-one-out cross-validation) are shown in Table 2. We can notice that the gene sets constructed using our novel method performed equally well as the gene sets based on fully-coupled fluxes. Interestingly, our gene sets contained about half the number of genes as compared to FCFs and despite that they were able to perform equally well.

7 Conclusions and Future Work

In this paper we have introduced a novel relational learning system capable to work efficiently with combinations of relational and numerical data. The experiments with real-world gene-expression and proteomics data gave us some very promising results. Furthermore, there are other possible applications of Gaussian logic in predictive classification settings which were not discussed in this paper. For example, finding patterns that generally correspond to highly correlated sets (not necessarily correlated with the class) of genes may have applications with group-lasso based classification approaches [12].

Acknowledgement. We thank the anonymous reviewers for their very valuable comments. This work was supported by the Czech Grant Agency through project 103/10/ 1875 *Learning from Theories* and project 103/11/2170 *Transferring ILP techniques to SRL.*

Appendix

In this appendix, we describe technical details concerning estimation of μ-vectors and $\mathbf{\Sigma}$-matrices.

Proof of Lemma 1

Let us suppose, for contradiction, that the assumptions of the lemma are satisfied, $\delta > 0$ and that

$$\delta = \lim_{n\to\infty} Pr\left(|E^* - \frac{1}{|\mathcal{E}_n|}\sum_{\widehat{E}_i\in\mathcal{E}_n}\widehat{E}_n^i| > \epsilon\right) \leq \lim_{n\to\infty} Pr\left(\frac{1}{|\mathcal{E}_n|}\sum_{\widehat{E}_i\in\mathcal{E}_n}|E^* - \widehat{E}_n^i| > \epsilon\right)$$

From this we have

$$\lim_{n\to\infty} \mathbf{E}\left(\frac{1}{|\mathcal{E}_n|}\sum_{\widehat{E}_i\in\mathcal{E}_n}|E^* - \widehat{E}_n^i|\right) \geq \delta\cdot\epsilon > 0$$

but

$$\lim_{n\to\infty} \mathbf{E}\left(\frac{1}{|\mathcal{E}_n|}\sum_{\widehat{E}_i\in\mathcal{E}_n}|E^* - \widehat{E}_n^i|\right) = \lim_{n\to\infty}\left(\frac{1}{|\mathcal{E}_n|}\sum_{\widehat{E}_i\in\mathcal{E}_n}\mathbf{E}|E^* - \widehat{E}_n^i|\right) =$$
$$= \lim_{n\to\infty}\left(\frac{1}{|\mathcal{E}_n|}\cdot|\mathcal{E}_n|\cdot\mathbf{E}|E^* - \widehat{E}_n^i|\right) = \lim_{n\to\infty}\left(\mathbf{E}|E^* - \widehat{E}_n^i|\right) = 0$$

(where the last equality results from the convergence in mean of the individual estimators) which is a contradiction. The only remaining possibility would be that the limit does not exist but then we can select a subsequence of $\mathcal{E}_i$ which has a non-zero limit and again derive the contradiction as before. □

Parameter Estimation

In this section we describe an efficient algorithm for estimation of μ-vectors and $\mathbf{\Sigma}$-matrices (polynomial in the combined size of a feature and an example) for the class of *tree-like conjunctive features*. Algorithms for computing quantities related to $\boldsymbol{\mu}(\varphi, e_i)$ for *tree-like features* have already been described in literature on relational aggregation [13]. However, there has been no prior work concerned with tractable computation of $\mathbf{\Sigma}(\varphi, e_i)$ or any similar quantity.

Definition 1 (Tree-like conjunction). *A first-order conjunction without quantifications C is* tree-like *if the iteration of the following rules on C produces the empty conjunction: (i) Remove an atom which contains fewer than 2 variables. (ii) Remove a variable which is contained in at most one atom.*

Intuitively, a tree-like conjunction can be imagined as a tree with the exception that whereas trees are graphs, conjunctions correspond in general to hypergraphs.

We start by some auxiliary definitions. Let φ be a tree-like feature. Let us suppose that $s_1, s_2, \ldots, s_k$ is a sequence of steps of the reduction procedure from Definition 1 which produces an empty feature from φ. Let $\lhd$ be an order on the atoms of φ such that if an atom a_1 disappeared before an atom a_2 during the reduction process then $a_1 \lhd a_2$. Then we say that $\lhd$ is a topological ordering of φ's atoms. Let $A \subseteq \varphi$ be a maximal set of atoms having a variable v in common. We say that $a \in A$ is a parent of atoms from $A \setminus \{a\}$ if for each $x \in A \setminus \{a\}$ it holds $x \lhd a$ (we also say that the atoms in $A \setminus \{a\}$ are a's children). An atom a is called root if it has no parents w.r.t. $\lhd$, it is called a leaf if it has no children w.r.t. $\lhd$.

We will use $(C, v) \in Children(\varphi, \lhd)$ for the set of all features φ_C with roots equal to children of φ (w.r.t. $\lhd$) together with the respective shared variables v. Similarly, we will use $C \in Children(\varphi, v, \lhd)$ for the set of all features φ_C with roots equal to children of φ (w.r.t. $\lhd$) sharing the variable v with φ's root. We will also use $arg_i(a)$ to identify the term appearing in the i-th argument of a. Next, to denote the set of all arguments of $a = atom(a_1, a_2, \ldots, a_k)$ and their positions in the atom we will use $(a_i, i) \in args(a)$. Finally, we define the *input* variable of a feature φ contained in some bigger feature ψ (denoted by $inp(\varphi, \psi, \lhd)$) as the variable which is shared by $root(\varphi, \lhd)$ with its parent in ψ. When a is a ground atom such that $root(\varphi, \lhd)\theta = a$ then we define *input* operator $inp(a, \varphi, \psi, \lhd)$ which will give us $inp(\varphi, \psi, \lhd)\theta$ (i.e. the term from the argument corresponding to the input variable in φ). If $\varphi = \psi$ then $inp(a, \varphi, \psi, \lhd) = inp(\varphi, \psi, \lhd) = \emptyset$[5]. We say that ψ is a sub-feature (of φ) if $\psi \subseteq \varphi$ and ψ and $\varphi \setminus \psi$ are both connected features. The parameter estimation algorithm will use two auxiliary algorithms for computing so-called *domains* and *term-domains* of features which are defined as follows.

Definition 2 (Domain, term-domain). *Let e be an example. Let φ be a tree-like feature, $\lhd$ be a topological ordering of φ's atoms. Then we say that a set of atoms $A \subseteq e$ is domain of φ w.r.t. e (denoted by $A = \mathcal{D}(\varphi, e, \lhd)$) if A contains all ground atoms $a = root(\varphi, \lhd)\theta$ such that $e \models \varphi\theta$. If φ is contained in a bigger feature ψ then we can also define its term-domain as $\mathcal{D}_T(\varphi, \psi, e, \lhd) = \{inp(\varphi, \psi, \lhd)\theta | e \models \varphi\theta\}$.*

Let $e = a(a, b), a(a, c), b(b)$ be an example and let $\varphi = a(X, Y), b(Y)$ and $\psi = a(X, Y), a(X, Z), b(Y)$ be features. Let $b(Y) \lhd a(X, Y)$. Then $\mathcal{D}(\varphi, e, \lhd) = \{a(a, b)\}$ and $\mathcal{D}_T(\varphi, \psi, e, \lhd) = \{a\}$.

[5] Here, the empty set is used as a *dummy* input. It does not mean that the returned value of the *input* functions would be a set in general.

Algorithms for computing domains and term-domains have been described in [16] and they also correspond to well-known algorithms for answering acyclic conjunctive queries [22]. These algorithms run in time polynomial in $|\varphi|$ and $|e|$. We note that these algorithms for computing domains of tree-like features compute not only domains corresponding to the roots of the given features but also domains corresponding to all the sub-features during one pass over a given feature. In the pseudocode of the parameter-estimation algorithms we will *call* the procedure for computing domains using $\mathcal{D}(\varphi, e, \lhd, T)$ where φ is the feature and e is the example for which we want to compute the domain, $\lhd$ is a topological ordering of φ's atoms and T is a table in which domains of all φ's sub-features should be stored. Similarly, we will *call* the procedure for computing term-domains using $\mathcal{D}_T(\varphi, \psi, e, \lhd, T)$ where again φ and e are the feature and the example for which we want to compute the term-domain, ψ is a feature containing φ (recall the definition of term-domain), $\lhd$ is a topological ordering of ψ's atoms and T is a table in which the computed term-domains of φ's sub-features should be stored.

Now, we can proceed further to computation of the parameters $\boldsymbol{\mu}(\varphi, e)$ and $\boldsymbol{\Sigma}(\varphi, e)$. By *sample parameters* we will mean a 5-tuple $(x, \widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}}, n, \gamma)$. Here x can be either an empty set, an atom or a term, $\boldsymbol{\mu}$ is a vector, $\boldsymbol{\Sigma}$ is a matrix and n is a natural number. The parameter γ is an ordered list of the distinguished variables $R_i \in vars(\varphi)$. Next, we define a concatenation operation for combining sample parameters.

Definition 3 (Concatenation operator). *Let $A = (x, \boldsymbol{\mu}_A, \boldsymbol{\Sigma}_A, n_A, \gamma_A)$ and $B = (y, \boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B, n_B, \gamma_B)$ be sample parameters. Then we define $A \otimes B$ as*

$$A \otimes B = \left(x, \left[\, \boldsymbol{\mu}_A^T \; \boldsymbol{\mu}_B^T \,\right]^T, \boldsymbol{\Sigma}_{AB}, n_A \cdot n_B, \gamma_A \cup \gamma_B\right)$$

where $\gamma_A \cup \gamma_B$ denotes concatenation of the lists γ_A and γ_B and $\boldsymbol{\Sigma}_{AB}$ is a block-diagonal matrix

$$\boldsymbol{\Sigma}_{AB} = \begin{bmatrix} \boldsymbol{\Sigma}_A & 0 \\ 0 & \boldsymbol{\Sigma}_B \end{bmatrix}$$

Definition 4 (Combination operator). *Let $\varphi \subseteq \psi$ be features and $\lhd$ a topological ordering of ψ's atoms. Let γ be a list of distinguished variables $R_i \in vars(\varphi)$. Let $A = (x, \boldsymbol{\mu}_A, \boldsymbol{\Sigma}_A, n_A, \gamma)$ and $B = (y, \boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B, n_B, \gamma)$ be sample parameters where x and y are logic atoms such that $inp(x, \varphi, \psi, \lhd) = inp(y, \varphi, \psi, \lhd)$. Then we define $A \oplus_{\lhd}^{\varphi,\psi} B$ as $A \oplus_{\lhd}^{\varphi,\psi} B = (inp(x, \varphi, \psi, \lhd), \boldsymbol{\mu}_{AB}, \boldsymbol{\Sigma}_{AB}, n_A + n_B, \gamma)$ where $\boldsymbol{\mu}_{AB} = \frac{1}{n_A + n_B}\left(n_A \cdot \boldsymbol{\mu}_A + n_B \cdot \boldsymbol{\mu}_B\right)$ and*

$$\boldsymbol{\Sigma}_{A,B} = \frac{1}{n_A + n_B}\left(n_A \cdot (\boldsymbol{\Sigma}_A + \boldsymbol{\mu}_A \cdot \boldsymbol{\mu}_A^T) + n_B \cdot (\boldsymbol{\Sigma}_B + \boldsymbol{\mu}_B \cdot \boldsymbol{\mu}_B^T)\right) - \boldsymbol{\mu}_{AB} \cdot \boldsymbol{\mu}_{AB}^T$$

Let us note that $\oplus_{\lhd}^{\varphi,\psi}$ is a commutative and associative operation which is also implicitly used in the following definition.

Definition 5 (Combination operator). *Let $\varphi \subseteq \psi$ be features and $\lhd$ a topological ordering of φ's atoms. Next, let γ be a list of distinguished variables $R_i \in$*

Algorithm 1. An algorithm for computing $\boldsymbol{\mu}(\varphi, e)$ and $\boldsymbol{\Sigma}(\varphi, e)$ for connected φ

Procedure: $\mu\sigma(\varphi, e = (H, \boldsymbol{\theta}), \lhd)$
1: $T \leftarrow []$
2: $\mathcal{D}(\varphi, e, \lhd, T)$ /* *This fills values into the table* T */
3: **return** $\bigoplus_{\lhd}^{\varphi,\varphi} \mu\sigma'(\varphi, \varphi, e, \lhd, T)$

Procedure: $\mu\sigma'(\varphi, \psi, e = (H, \boldsymbol{\theta}), \lhd, T)$
1: $SP \leftarrow []$ /* SP *is an associative array of sample parameters* */
2: $D_\varphi \leftarrow T[\varphi]$ /* D_φ *is domain of* φ */
3: **for** $\forall a \in D_\varphi$ **do**
4: $SP[a] \leftarrow \{a, \boldsymbol{\theta}_{I(a)}, |I(a)| \times |I(a)|$ zero matrix$, 1, \mathcal{R}_a\}$ /* where $\mathcal{R}_a$ is a list of the distinguished variables contained in $root(\varphi)$ */
5: **end for**
6: **for** $(\varphi_C, v) \in Children(\varphi, \lhd)$ **do**
7: $SP_{\varphi_C} \leftarrow \bigoplus_{\lhd}^{\varphi_C,\varphi} \mu\sigma'(\varphi_C, \varphi, e, \lhd)$
8: **for** $\forall a \in D_\varphi$ **do**
9: $SP[a] \leftarrow SP[a] \otimes SP_{\varphi_C}[v\vartheta]$ where $root(\varphi, \lhd)\vartheta = a$
10: **end for**
11: **end for**
12: **return** SP

$vars(\varphi)$. *Let* $\mathcal{X} = \{(x_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, n_1, \gamma), \ldots, (x_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, n_k, \gamma)\}$. *Next, let* $\mathcal{X}[t]$ *denote the set of all sample parameters* $(x, \ldots) \in \mathcal{X}$ *for which* $inp(x, \varphi, \psi, \lhd) = t$. *Then* $\bigoplus_{\lhd}^{\varphi,\psi} \mathcal{X}$ *is defined as follows:*

$$\bigoplus_{\lhd}^{\varphi,\psi} \mathcal{X} = \{(y_1, \boldsymbol{\mu}_{y_1}, \boldsymbol{\Sigma}_{y_1}, n_{y_1}, \gamma), \ldots, (y_m, \boldsymbol{\mu}_{y_m}, \boldsymbol{\Sigma}_{y_m}, n_{y_m}, \gamma)\}$$

where $(y_i, \boldsymbol{\mu}_{y_i}, \boldsymbol{\Sigma}_{y_i}, n_{y_i}, \gamma) = x_1 \oplus_{\lhd}^{\varphi,\psi} x_2 \oplus_{\lhd}^{\varphi,\psi} \cdots \oplus_{\lhd}^{\varphi,\psi} x_o$ *for* $\{x_1, \ldots, x_o\} = \mathcal{X}[y_i]$.

The basic ideas underlying Algorithm 1 are summarized by the next two observations.

Observation 1. *Let* $\varphi = \psi \cup C_1 \cup \cdots \cup C_n$ *be a feature where each* C_i *is a sub-feature of* φ *and* $\psi \cap C_i = \emptyset$ *and* $C_i \cap C_j = \emptyset$ *for* $i \neq j$. *Let* ϑ *be a substitution affecting only variables* $v \in vars(\psi)$ *and guaranteeing that* $\psi\vartheta$ *will be ground and it will hold* $e \models \varphi\vartheta$ *where* $e = (H, \boldsymbol{\theta})$ *is an example. Then* $\boldsymbol{\mu}(\varphi\vartheta, e)$ *and* $\boldsymbol{\Sigma}(\varphi\vartheta, e)$ *and the number of samples* $m = |\mathcal{S}(\varphi\vartheta, e)|$ *are given by the sample parameters* A *(up to reordering of random variables) computed as*

$$A = (\psi\vartheta, \boldsymbol{\theta}_I(\vartheta), n \times n \text{ zero matrix}, 1, \mathcal{R}_\psi) \otimes \ldots$$
$$\otimes \mu\sigma(C_1\vartheta, e, \lhd\vartheta) \otimes \mu\sigma(C_2\vartheta, e, \lhd\vartheta) \otimes \cdots \otimes \mu\sigma(C_k\vartheta, e, \lhd\vartheta)$$

(where $n = |I(\psi)|$ *and* $\mathcal{R}_\psi$ *is a list of the distinguished variables* $R_i \in vars(\psi)$*) provided that* $\mu\sigma(C_i\vartheta, e, \lhd\vartheta)$ *are correct sample parameters.*

Let us look more closely at what $\mu\sigma(C_i\vartheta, e, \lhd\vartheta)$ is. First, we can notice that $C_i\vartheta$ is a sub-feature of φ which differs from C_i only by the fact that it has its input variable ($inp(C_i, \varphi, \lhd)$) grounded by ϑ. Therefore $\mu\sigma(C_i\vartheta, e, \lhd\vartheta)$ can be also obtained from the set $\bigoplus_{\lhd}^{C_i,\varphi} \mu\sigma'(C_i, \varphi, e, \lhd, T)$ (where the argument T is a table containing pre-computed domains). The next observation, in turn,

shows that what $\bigoplus_{\lhd}^{C_i,\varphi} \mu\sigma'(C_i, \varphi, e, \lhd, T)$ contains are the sample parameters corresponding to $\boldsymbol{\mu}(C_i\vartheta, e)$, $\boldsymbol{\Sigma}(C_i\vartheta, e)$ and the number of samples $|\mathcal{S}(C_i\vartheta, e)|$ for all substitutions ϑ grounding only the input argument of C_i such that $e \models C_i\vartheta$.

Observation 2. *Let $\varphi \subseteq \psi$ be features and let $e = (H, \boldsymbol{\theta})$ be an example. Let $\vartheta : inp(\varphi, \psi, \lhd) \to \mathcal{D}_T(\varphi, \psi, e, \lhd)$ be a substitution. Then $\boldsymbol{\mu}(\varphi\vartheta, e)$, $\boldsymbol{\Sigma}(\varphi\vartheta, e)$, $n_{\varphi\vartheta} = |\mathcal{S}(\varphi\vartheta, e)|$ are contained in*

$$(inp(\varphi, \psi, \lhd)\vartheta, \boldsymbol{\mu}(\varphi\vartheta, e), \boldsymbol{\Sigma}(\varphi\vartheta, e), n, \gamma) \in \bigoplus_{\lhd}^{\varphi,\psi} \mu\sigma'(\varphi, \psi, e, \lhd, T)$$

(where T is a table with pre-computed domains of sub-features of φ) provided that $\mu\sigma'(\varphi, \psi, e, \lhd, T)$ are correct sample parameters.

References

1. Ahmad, S., Sarai, A.: Moment-based prediction of dna-binding proteins. Journal of Molecular Biology 341(1), 65–71 (2004)
2. Best, C.J.M., et al.: Molecular alterations in primary prostate cancer after androgen ablation therapy. Clin. Cancer Res. 11(19 Pt 1), 6823–6834 (2005)
3. Bhardwaj, N., Langlois, R.E., Zhao, G., Lu, H.: Kernel-based machine learning protocol for predicting DNA-binding proteins. Nucleic Acids Research 33(20), 6486–6493
4. Burczynski, M.E., et al.: Molecular classification of crohns disease and ulcerative colitis patients using transcriptional profiles in peripheral blood mononuclear cells. J. Mol. Diagn. 8(1), 51–61 (2006)
5. Dahia, P.L.M., et al.: A hif1alpha regulatory loop links hypoxia and mitochondrial signals in pheochromocytomas. PLoS Genet. 1(1), 72–80 (2005)
6. Domingos, P., Kok, S., Lowd, D., Poon, H., Richardson, M., Singla, P.: Markov logic. In: De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S.H. (eds.) Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911, pp. 92–117. Springer, Heidelberg (2008)
7. Freije, W.A., et al.: Gene expression profiling of gliomas strongly predicts survival. Cancer Res. 64(18), 6503–6510 (2004)
8. Gashaw, I., et al.: Gene signatures of testicular seminoma with emphasis on expression of ets variant gene 4. Cell Mol. Life Sci. 62(19-20), 2359–2368 (2005)
9. Gordon, G.J.: Transcriptional profiling of mesothelioma using microarrays. Lung Cancer 49(suppl.1), S99–S103 (2005)
10. Higham, N.J.: Computing the nearest correlation matrix - a problem from finance. IMA Journal of Numerical Analysis, 329–343 (2002)
11. Holec, M., Železný, F., Kléma, J., Tolar, J.: Integrating multiple-platform expression data through gene set features. In: Măndoiu, I., Narasimhan, G., Zhang, Y. (eds.) ISBRA 2009. LNCS, vol. 5542, pp. 5–17. Springer, Heidelberg (2009)
12. Jacob, L., Obozinski, G., Vert, J.-P.: Group lasso with overlap and graph lasso. In: Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, pp. 433–440. ACM, New York (2009)
13. Jakl, M., Pichler, R., Rümmele, S., Woltran, S.: Fast counting with bounded treewidth. In: Cervesato, I., Veith, H., Voronkov, A. (eds.) LPAR 2008. LNCS (LNAI), vol. 5330, pp. 436–450. Springer, Heidelberg (2008)

14. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., Hattori, M.: The kegg resource for deciphering the genome. Nucleic Acids Research 1 (2004)
15. Kuriakose, M.A., et al.: Selection and validation of differentially expressed genes in head and neck cancer. Cell Mol. Life Sci. 61(11), 1372–1383 (2004)
16. Kuželka, O., Železný, F.: Block-wise construction of tree-like relational features with monotone reducibility and redundancy. Machine Learning, online first (2010), doi:10.1007 / s10994-010-5208-5
17. Scherzer, C.R., et al.: Molecular markers of early parkinsons disease based on gene expression in blood. Proc. Natl. Acad Sci. US A 104(3), 955–960 (2007)
18. Stawiski, E.W., Gregoret, L.M., Mandel-Gutfreund, Y.: Annotating nucleic acid-binding function based on protein structure. Journal of Molecular Biology 326(4), 1065–1079 (2003)
19. Szilágyi, A., Skolnick, J.: Efficient prediction of nucleic acid binding function from low-resolution protein structures. Journal of Molecular Biology 358(3), 922–933 (2006)
20. Tsuchiya, Y., Kinoshita, K., Nakamura, H.: Structure-based prediction of dna-binding sites on proteins using the empirical preference of electrostatic potential and the shape of molecular surfaces. Proteins: Structure, Function, and Bioinformatics 55(4), 885–894 (2004)
21. Wang, J., Domingos, P.: Hybrid markov logic networks. In: Proceedings of the 23rd national conference on Artificial intelligence, vol. 2. AAAI Press, Menlo Park (2008)
22. Yannakakis, M.: Algorithms for acyclic database schemes. In: International Conference on Very Large Data Bases (VLDB 1981), pp. 82–94 (1981)

Toward a Fair Review-Management System

Theodoros Lappas[1,⋆] and Evimaria Terzi[2]

[1] UC Riverside
tlappas@cs.ucr.edu
[2] Boston University
evimaria@cs.bu.edu

Abstract. Item reviews are a valuable source of information for potential buyers, who are looking for information on a product's attributes before making a purchase decision. This search of information is often hindered by overwhelming numbers of available reviews, as well as low-quality and noisy content. While a significant amount of research has been devoted to filtering and organizing review corpora toward the benefit of the buyers, a crucial part of the reviewing process has been overlooked: *reviewer satisfaction*. As in every content-based system, the content-generators, in this case the reviewers, serve as the driving force. Therefore, keeping the reviewers satisfied and motivated to continue submitting high-quality content is essential. In this paper, we propose a system that helps potential buyers by focusing on high-quality and informative reviews, while keeping reviewers content and motivated.

1 Introduction

Item reviews are among the most prominent instances of opinionated text found on the Web. By authoring a review, a reviewer can share his experience and express opinions on the advantages and disadvantages of an item's features. The accumulated corpus of reviews can then serve as a valuable source of information for interested users and potential buyers. Due to their immense popularity and important role in the modern e-commerce model, reviews have been the focus of numerous interesting research problems. The challenges that emerge in a real review-hosting system can be grouped in the following two categories:

Volume and Redundancy: Users are often faced with an overwhelming volume of reviews. As of April 2011, Amazon.com hosts almost 21,000 reviews on the popular Kindle reading device. Clearly, it is impractical for a user to go through hundreds, or even thousands, of reviews in order to obtain the information she is looking for. Moreover, a large portion of the reviews are often redundant, expressing the same opinions on the same attributes. A number of approaches have been proposed to address these challenges, mainly focusing on summarization [6,21,12] and search-based methods [11].

Review Quality: As is the case in every system that hosts user-generated content, the quality of the submitted information is a primary issue. Several

⋆ This work was done while the author was visiting Boston University.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 293–309, 2011.

notions of review quality have been proposed and evaluated in the relevant literature [12,11,14]. The quality of a review is based on the volume and validity of the information it conveys, as well as its readability and structural features.

While significant contributions have been made toward addressing these challenges, we identify two major parts of the review-management process that are still lacking: (a) review presentation and visibility and (b) reviewer motivation and utilization. Next, we discuss these two in more detail.

Review Presentation: In the context of a review-hosting website, the review presentation component determines the visibility of each review in terms of both time and placement on the site's interface. Since the attention span of the users is limited, decisions about which reviews to show can have a major impact on their browsing experience and overall satisfaction.

In major contemporary review portals, reviews are usually sorted based on the date of submission. This approach clearly does not address any of the challenges discussed above. Alternative sorting methods are based on user feedback. On sites like `Amazon.com` and `Yelp.com` users can rate the *helpfulness* of a review. These ratings can be then used for ranking purposes. Further, reviewers inherit the ratings of their authored reviews, providing a measure of expertise. This information can then be considered for future reviews. Recently, `Yelp.com` has introduced a method called YelpSort. The website claims that this measure evaluates reviews based on "recency, ratings and other review quality factors". Even though this is a step toward the right direction, no details on the actual computation are provided. While user ratings can be a useful source of information, they also suffer from significant drawbacks [12]. For example, reviews that already have many votes are more likely to attract even more, since they are granted increased visibility (e.g. ranked higher) by the system. In addition, older reviews have more time to accumulate helpfulness votes and tend to overwhelm new (and potentially superior) reviews.

Reviewer Motivation and Utilization: In existing review-hosting portals, reviewers have no interaction with the system. This leads to two significant shortcomings:

1. *Underutilization of expertise:* Writing a review is an expression of the reviewer's motivation to comment on a particular subset of the item's attributes. However, even though the same person may be perfectly capable to comment on more attributes, he may refrain from doing so due to negligence or lack of motivation. In other words, existing systems fail to get the most out of the reviewer and thus deprive potential customers from informative content.
2. *Lack of motivation:* Fully understanding the process that motivates certain users to review certain items is a non-trivial task. Possible causes include a genuine desire to help others, frustration or excitement due to the reviewed item, the desire to influence others and gain acknowledgment via positive ratings, or simply the need to express one's self. In any case, it is safe to

say that, when a user submits a review on a public website, he does so in anticipation that his opinions will be read by others. Hence, visibility is the primary means that the system can utilize in order to motivate reviewers. For this to work in practice, there needs to be a clear connection between the usefulness of the submitted review and the visibility it is granted. However, current systems either completely disregard the satisfaction of the reviewers, or try to indirectly motivate them with the promise of user-submitted helpfulness votes. We have already discussed the biases that ail such mechanisms earlier in the paper.

1.1 Contribution

In this paper, we present practical and theoretically grounded solutions for addressing the above shortcomings. More specifically, we present a framework that keeps the users informed and the reviewers motivated to contribute high-quality content. Our system guarantees that users are presented with a compact set of high-quality reviews that cover *all* the attributes of the item of their interest. We also guarantee reviewer satisfaction by proposing a mechanism which periodically *shuffles* the existing set of reviews, instead of statically ranking them with respect to certain criteria. The design of the shuffling mechanism is such that the chances of the review to be brought into the spotlight are proportional to its usefulness to the user. Finally, we present a mechanism for suggesting to reviewers how to extend their reviews in order to gain more visibility.

1.2 Roadmap

The rest of the paper is organized as follows: we review the related work in Section 2. After a brief introduction of our notation in Section 3, we present our methods for review shuffling and user motivation and utilization in Sections 4 and 5. In Section 6 we provide a set of experiments that demonstrate the practical utility of our approach. We conclude the paper in Section 7.

2 Related Work

Our work is the first to propose a complete review management system that balances the satisfaction of customers, as well as reviewers. Nonetheless, our methodology has ties to areas relevant to the domain of item reviews.

Review Quality: The problem of formalizing and evaluating the quality of a review has attracted considerable attention. A large volume of work has been devoted to evaluating the *helpfulness* of reviews [14,20], typically formulating the problem as one of classification or regression. Jindal and Liu [8] focus on the detection of spam (e.g. duplicate reviews). Liu and Cao [12], formulate the problem as a binary classification task, assigning a quality rating of "high" or "low" to reviews. In recent work, Lu et al. [13] discuss how mining information from social networks can be used toward the evaluation of review quality. A

different notion of quality deals with the readability of a review, as defined by as its structural characteristics. The Flesch Reading EASE [9] is indicative of this line of work. Although our framework includes a component for the evaluation of review quality, our ultimate goal is to build a system that balances user and reviewer satisfaction.

Attribute Extraction and Opinion Mining: Given a review corpus on an item, opinion mining [4,7,16,17], looks for the attributes of the item that are evaluated in each review, as well as the respective opinions expressed on each attribute. For our experiments, we implemented the technique proposed by Hu and Liu [7] for attribute extraction.

Review Management: The accumulation of overwhelming volumes of online reviews has created the need for methods to manage and present such data. A relevant field is that of opinion summarization [10,12,21], in which the review corpus is processed to produce a statistical summary with information on the distribution of positive and negative opinions on the attributes of the item. Other relevant approaches [11,22] propose methods for finding a compact and informative set of reviews. Finally, the Information Systems community has explored the problem of review placement and its effect on customers [18]. Contrary to our own approach, none of these methods take into consideration the motivation and satisfaction of the reviewers. We present a framework that helps customers deal with the large number of reviews, while keeping reviewers motivated to submit high-quality and informative content.

3 Notation

We use A to denote the set of attributes associated with the given item. We also use R to denote the set of all reviews that have been written for the item. We assume that $|A| = m$ and $|R| = n$. Every review $r \in R$ is represented as a subset of the item's attributes; that is, $r \subseteq A$. We assume that every reviewer writes a single review for each item, and therefore every review r uniquely identifies its author.

4 Spotlight Shuffling

In this section, we present our method for selecting the set of reviews to be shown to the users. We call the set of reviews shown at any point in time to the visitors of the host site, the *spotlight set*. If R is the set of all the reviews of an item, then the spotlight set S is a subset of the reviews from R (i.e., $S \subseteq R$). Our mechanism for review selection is based on creating different spotlight sets at different points in time. Therefore, the output of our method is a sequence of N spotlight sets, $\mathcal{S} = \{S_1, S_2, \ldots, S_N\}$. Ideally, we would like the following properties for the spotlight sequence.

- **Attribute coverage:** each spotlight set needs to cover *all* the attributes of the item, in order to provide a thorough presentation to the interested user.
- **Review Quality:** each spotlight set has to include only high-quality reviews, that are both informative and easy to read.
- **Fair spotlight share:** each eligible review needs to have a fair chance of inclusion in the spotlight sequence, according to the information it conveys.
- **Compactness:** every spotlight set of $\mathcal{S}$ needs to be compact, so that users can read through it in a reasonable amount of time.

In the remaining of this section, we identify concepts and methods that will allow us to produce spotlight sequences that satisfy the above requirements.

4.1 Attribute Coverage

We first define the notion of the *cover*, which is central to our analysis:

Definition 1. *Given the corpus of reviews R on an item with attributes A, a a subset of the reviews $R' \subseteq R$ is a* cover *of the item if every attribute of A is evaluated in at least one of the reviews in R'. That is, $\cup_{r \in R'} r = A$.*

In order to satisfy the attribute-coverage requirement we require every set in the spotlight sequence to be a cover. In fact, for the rest of the discussion, we will use the terms *spotlight set* and *cover* interchangeably.

4.2 Review Quality

The second requirement demands that all the spotlight sets in the sequence consist only of high-quality reviews. That is, if $q(r)$ is a numeric measure of the quality of the review r, then we say that a spotlight set S is *high-quality* if for every $r \in S$ $q(r) > \tau$. Here, τ is a minimum threshold imposed on the quality of the reviews in $\mathcal{S}$. The next step is to find an intuitive definition that accurately captures the quality of a review. Our methodology is compatible with any approach for review-quality evaluation. We refer the reader Section 2 for a thorough overview on related work. In our implementation, we adopt the *Flesch Readability Ease* (FRE) formula [9] . Formally, the FRE score of a given (English) review r is defined as:

$$FRE(r) = 206.835 - 1.015 \times \frac{words(r)}{sents(r)} - 84.6 \times \frac{syllables(r)}{words(r)},$$

where $words(r)$, $sents(r)$ and $syllables(r)$ denote the number of words, sentences and syllables in d, respectively. This is very popular formula, the weights of which have been derived by means of regression on training data. The formula yields numbers from 0 to 100, expressing the range from "very difficult" to "very easy", and is meant to be used for measuring the readability of texts addressed to adult language users. In our experiments, we discard reviews that score less than $\tau = 60$, a typically used threshold for FRE [9]. We choose FRE for its popularity and its use as a standard for readability by many organizations (e.g., by the U.S. Department of Defense).

4.3 Fair Spotlight Share

For a given item, one can pick a high-quality spotlight set S by simply appending to S high-quality reviews until all the attributes in A are covered. Of course, one can see that there are exponentially many distinct high-quality spotlight sets. For the rest of the discussion, we denote the complete collection of spotlight sets by $\mathcal{C}$. The question that we consider here is: "how can we *fairly* construct a spotlight sequence by picking spotlight sets from $\mathcal{C}$?".

In order to address this problem, we propose a shuffling scheme that constructs a spotlight sequence $\mathcal{S}$ in a fair way that ensures the satisfaction of both customers and reviewers: customers see a compact and informative set of reviews, while each reviewer receives a *fair* share of participation in the spotlight sequence. Clearly, the design of the shuffling method largely depends on the definition of "fair participation". In a fair shuffling scheme, valuable reviews should be rewarded. Intuitively, a review is valuable if it evaluates a large number of attributes or if it comments on significant attributes that are overlooked by the majority of the reviewers. As an example, consider an expert review that provides insightful opinions on attributes that are too hard for the average reviewer to evaluate (e.g. the advanced technical features of a laptop computer). Taking the above into consideration, we formalize the *spotlight privilege* of a given review r as follows:

Definition 2. *Let $\mathcal{C}$ be the collection of all covers of a particular item. Then, the spotlight privilege $p(r)$ of a given review r is equal to the number of spotlight sets in $\mathcal{C}$ that include r:*

$$p(r) = |\{S \in \mathcal{C} \mid r \in S\}|.$$

Conceptually, the more spotlight sets a review participates in, the higher its spotlight privilege. Thus, in the shuffling scheme, every time we need to extend the spotlight sequence, it is sufficient to choose a new spotlight set from $\mathcal{C}$, uniformly at random. If we repeat this sampling process for an appropriate number of times, the *expected* number of times a review r is included in the spotlight sequence will converge to $p(r)$.

The Spotlight Shuffling Algorithm: Given the above discussion, the next issue is how to sample uniformly at random from the collection of covers $\mathcal{C}$. An intuitive algorithm is the following: pick random subsets $R' \subseteq R$ and check whether R' is a spotlight set. If it is, then it is presented to the users and the algorithm proceeds in the same fashion to pick the spotlight set to show in the next timestamp. Although this algorithm is both natural and simple, its time complexity is exponential. The reason for this is that $\mathcal{C}$ can be very small compared to the 2^n possible subsets of R. Therefore, the algorithm needs to pick many subsets before actually finding a valid spotlight set.

An alternative is to explicitly construct spotlight sets from $\mathcal{C}$. Such an algorithm could initialize the spotlight sequence $\mathcal{S}$ with the spotlight set $S_1 = R$ (i.e. the full set of reviews). Spotset S_i is then created from spotset S_{i-1} by removing from S_i one review at a time, as long as the remaining set is still a spotlight

Algorithm 1. The `ImportanceSampling` algorithm.

Input: Set of minimal covers $\mathcal{M}$, number of desired samples N.
Output: Spotlight sequence $\mathcal{S}$ of length N .

$\mathcal{S} \leftarrow \emptyset$
while $|\mathcal{S}| < N$ **do**
  pick M_i from $\mathcal{M}$ with probability $\frac{2^{n-|M_i|}}{\sum_{M \in \mathcal{M}} 2^{n-|M|}}$
  Generate a superset $S \in \mathcal{C}_i$ of M_i by appending each review $r \in R \setminus M_i$ with probability 1/2.
  Let i^* be the *Canonical Representative* of S.
  if $i^* = i$ **then** $\mathcal{S} = \mathcal{S} \cup \{S\}$
return $\mathcal{S}$

set. Although this algorithm guarantees the construction of many spotsets in polynomial time, it still does not solve our problem. This is because it does not guarantee uniform sampling from $\mathcal{C}$. In fact, by using this algorithm, several elements from $\mathcal{C}$ might never be discovered.

In order to address the problems discussed above, we employ *importance sampling* [15] in order to uniformly sample solutions from $\mathcal{C}$. Next, we discuss how the technique can be applied to our setting.

The `ImportanceSampling` Algorithm: The importance sampling technique requires as input the set of all *minimal* spotlight sets. Recall that a spotlight set S is minimal if it is not a proper super-set of any other possible spotlight set. In other words, every review in a minimal spotlight set S is the only review in S that covers at least one of the item's attributes. As before, we use $\mathcal{C}$ to denote the collection of all possible covers of the attribute-set A. We also use $\mathcal{M}$ to denote the set of all *minimal* spotlight sets. It is easy to see that every cover in $\mathcal{C}$ is a superset of at least one of the covers in $\mathcal{M}$. Even though $|\mathcal{M}| << |\mathcal{C}|$, computing $\mathcal{M}$ can still be a non-trivial task. For the sake of our analysis, we assume that $\mathcal{M}$ is available. Towards the end of this section, we show how to effectively sample from $\mathcal{M}$.

Let $M_i \in \mathcal{M}$ be a minimal cover and $\mathcal{C}_i$ be the collection of all supersets of M_i. In order for the technique to be applicable, the following three conditions need to be met [15]

1. We can can compute $|\mathcal{C}_i|$ in polynomial time.
2. We can sample uniformly at random from $\mathcal{C}_i$.
3. Given any subset of reviews $R' \subseteq R$, we can verify in polynomial time if $R' \in \mathcal{C}_i$.

In our case, all 3 conditions are satisfied: for (1) we have $|\mathcal{C}_i| = 2^{n-|M_i|}$. For (2), we can sample uniformly from the sets in $\mathcal{C}_i$ by appending each review in $R \setminus M_i$ with probability 1/2. For (3), given any $R' \subseteq R$, we can verify if R' is included in $\mathcal{C}_i$ by simply checking if R' is a superset of M_i.

The importance sampling technique is based on considering the multi-set $\mathcal{U} = \mathcal{C}_1 \uplus \ldots \uplus \mathcal{C}_{|\mathcal{M}|}$, where the elements of $\mathcal{U}$ are pairs of the form (C, i), corresponding

to a cover $C \in \mathcal{C}_i$. In other words, for every cover $C \in \mathcal{C}$, $\mathcal{U}$ contains as many copies of C as there are $\mathcal{C}_i$'s for which $C \in \mathcal{C}_i$. The multi-set $\mathcal{U}$ is then divided into equivalence classes, where each class contains all pairs (C, i) that correspond to the same cover $C \in \mathcal{C}$. A *single pair* (C, i) is defined to be the *canonical representation* of each class. The intuition is that, instead of sampling from the space of size 2^n that contains all possible subsets of reviews, we sample only from $\mathcal{U}$. This guarantees that the sampled subsets are indeed review covers. The algorithm is polynomial to the number of minimal reviews in $\mathcal{M}$ [15].

Selecting a Seed of Minimal Covers: So far, we have assumed that the set of minimal covers $\mathcal{M}$ is available. However, enumerating all the minimal covers[1] is computationally expensive. In fact, using any of the existing algorithms [2,3,5] we were unable to generate all minimal covers, even for very small datasets. Therefore, we propose to execute `ImportanceSampling` with a smaller *seed set*. We abuse notation and denote this set by $\mathcal{M}$. We generate elements of $\mathcal{M}$ by starting with set R and randomly removing elements until we reach a minimal solution. We repeat this process until we generate a seed of the desired size. Our intuition is that even a small seed is adequate to capture a subset for most of the solutions in $\mathcal{U}$. As we show in our experiments, $|\mathcal{M}| = O(n)$ is sufficient.

Covering Opinions: The standard definition of the spotlight set requires the covering of all the attributes of an item. An interesting alternative is to cover *opinions*. Given a review r, we first apply a method to extract the expressed opinions, where an opinion is defined as a mapping of an attribute to a positive or negative polarity. Then, a review with a positive opinion on attribute a_1 and a negative opinion on attribute a_2 would be represented by $\{a_1^+, a_2^-\}$, instead of $\{a_1, a_2\}$. It is important to note that our methodology is entirely compatible with this formulation (see also the example in Section 6.2).

4.4 Compactness

Compact spotlight sets that cover all the attributes of the item in a small number of reviews are preferred by the users, since they require much less effort to process. In order to give precedence to covers with a small number of reviews, we modify `ImportanceSampling` to give higher preference to small covers. For this, we associate with every cover S a weight $w(S)$ that only depends on the number of reviews in S. In our implementation, we use $w(S) = e^{-\lambda x_s}$, where $\lambda > 0$ and x_s is the size of S. In order to sample correctly from this weighted sample space, we need to modify the `ImportanceSampling` algorithm as follows: first, the minimal elements from $\mathcal{M}$ are sampled with probability proportional to the sum of the weights of their supersets. Second, the selected minimal set $M \in \mathcal{M}$ is extended as follows: Using an exponential prior, we select x_s with probability proportional to $e^{-\lambda x_s}$. Then, we form S by expanding M using x_s random reviews from $R \setminus M$. The rest of the algorithm proceeds as shown in the

[1] This is also known as the *transversal hypergraph* problem.

pseudocode in Algorithm 1. We explore the effect of the exponential prior in the experimental evaluation (Section 6).

5 Reviewer Motivation and Utilization

In this section, we discuss our methodology for improving the utilization of reviewer expertise. Our goal is to motivate reviewers to submit more high-quality content. Given a review r, we want to recommend to the author of the review a set of attributes $Q \subseteq \{A \setminus r\}$, such that the new extended review $r' = r \cup Q$ has better spotlight privileges. That is, r' appears in more spotlight sets in the spotlight sequence than the original review r.

Our attribute-recommendation system is based on the observation that the spotlight privileges of a review r increase with the number of reviews in R that it *dominates*. We say that a review r_i *dominates* a review r_j if $r_j \subseteq r_i$. That is, every attribute from A that is covered by r_j is also covered by r_i. We use $D(r)$ to denote the number of reviews from R that r dominates. Then, given a review r, our goal is to recommend its extension with attributes $Q \subseteq \{A \setminus r\}$ such that $D(r \cup Q)$ is maximized. Our attribute-recommendation system can be activated after the review is submitted (and parsed to extract attributes) or before (by asking the user to state the attributes he intends to review).

Clearly, if any reviewer was able to provide high-quality comments for all the attributes of a product, then the above problem would have a trivial solution: simply set $Q = A \setminus r$ and make the extended review r' dominate all the reviews in R. However, not all reviewers have the potential to comment on all attributes. Instead, the typical reviewer has the background and experience to comment on a subset of the item's attributes. For example, some reviewers might be in a better position to comment on the hardware parts of a laptop, while others may be able to provide a more comprehensive review on the accompanying software. To a certain extent, the ability of a reviewer to comment on a set of attributes is also encoded on his original review r. That is, the attributes that r covers are indicative of the attributes that this reviewer can comment on. Formally, we use $\Pr(a \mid r)$ to denote the probability that the author of review r is able to provide high-quality comments on a certain attribute $a \in A$. For now, we assume that these probabilities are given. Toward the end of the section, we discuss how we can compute them from the available data. Given a set of attributes $Q \subseteq A$, and the individual probabilities $\Pr(a \mid r), \forall a \in Q$, we compute the probability that a reviewer that wrote r can effectively evaluate the attributes in Q using the following independence assumption:

$$\Pr(Q \mid r) = \prod_{a \in Q} \Pr(a \mid r).$$

Given the above, the attribute-recommendation problem can now be formalized as follows.

*Problem 1 (*Attribute Recommendation - AR). Given an item with attributes A and a review $r \subseteq A$, find the set of attributes $Q \subseteq \{A \setminus r\}$ to recommend to the author of r such that

$$\textsc{Ed}(r \cup Q) = \Pr(Q \mid r) D(r \cup Q) = \left(\prod_{a \in Q} \Pr(a \mid r) \right) D(r \cup Q). \quad (1)$$

is maximized.

We call the quantity $\textsc{Ed}(r \cup Q)$ the *expected dominance* of the extended review $r' = r \cup Q$ and we refer to Problem 1 as the Attribute Recommendation problem (or AR for short). Note that the trivial solution $r' = A \setminus r$ is no longer optimal for the AR problem. In fact, as Q becomes larger, the $\Pr(Q \mid r)$ decreases. Also, as Q becomes larger, the second part of the objective (i.e., $D(r \cup Q)$) increases. This is because more reviews are getting dominated by $r \cup Q$.

Solving the AR Problem: Although we do not know the particular complexity of the AR problem, we know that the version of the problem where $\Pr(a \mid r) = 1$ for every $a \in A$ and the goal is to pick k attributes to form set Q such that $D(r \cup Q)$ is maximized is an NP-Complete problem.[2]

In our experiments, we deploy the following `Greedy` heuristic for the AR problem. At first, the algorithm sets $Q = \emptyset$. At iteration t, the algorithm forms set Q^t, by extending set Q^{t-1} with the single attribute that maximizes $\textsc{Ed}(r \cup Q^t) - \textsc{Ed}(r \cup Q^{t-1})$. The process repeats until none of the remaining attributes benefits the objective function. A single iteration of this algorithm has running time $O(nm)$. In the worst case scenario, the `Greedy` algorithm can have at most n iterations. In practice, however, the number is much smaller than n.

Computing the Probabilities $\Pr(a \mid r)$ for $a \in A$: Next, we discuss how we can compute $\Pr(a \mid r)$, i.e., the probability that the author of review r is able to provide high-quality comments on attribute a of an item. This is a challenging problem on its own. While we propose and experimentally evaluate a way to address this, we do not claim that we have exhaustively explored this particular problem. Instead, our purpose is to advocate the concept of attribute-recommendation for improved expertise utilization. In fact, our overall methodology is compatible with *any* method that can effectively compute $\Pr(a \mid r)$.

Consider the following motivating example. In the domain of digital cameras, the attributes *camera lens, digital zoom* and *optical zoom* are very often discussed in the same review, due to their close connection and interdependence. Intuitively, a person with particular interest for the zoom capabilities of a camera, is much more likely to also focus (and comment) on the camera's lens. We capture this intuition by computing the probability $\Pr(a \mid r)$ as follows:

$$\Pr(a \mid r) = \max_{s \in \mathcal{P}(r)} \frac{\text{freq}(s \cup a)}{\text{freq}(s)}, \quad (2)$$

where $\mathcal{P}(r)$ is the power set of the attributes in r and $\text{freq}(s)$ the number of reviews in the corpus that cover all the attributes in a given set s. Observe that Equation (2) is the *confidence* measure, as defined in the context of mining

[2] The reduction is from the well-known Set Cover problem.

association rules [1]. Hence, we can apply well-known rule-mining techniques to pre-compute the probabilities and reduce the processing time.

One can also take into account the reviewer's inherent ability and expertise on the item's domain. Then, if the reviewer has authored a number of well-received reviews on a topic, the probability that he is able to expand his initial review is elevated accordingly. This can be easily incorporated into Equation (2), by multiplying the left-hand side by a prior that captures the relevant expertise of the given review's author. The computation of such a factor falls within the area of expertise mining, and is orthogonal to our work. In our experiments, we assume a uniform prior distribution for all reviewers.

6 Experiments

In this section, we experimentally evaluate the methods presented in this paper. All the components of our system assume that we know the attributes that are discussed in every available review. To extract these attributes, we use the method proposed by Hu and Liu [7], which also mines opinions. An additional pass was made over the produced attribute-lists to verify their validity and also to address synonymy issues (e.g. *bathroom=restroom=toilet*).

6.1 Datasets

We use four item-review datasets provided by Lappas and Gunopulos [11]. The **GPS** and **TVS** datasets include the complete review corpora from `Amazon.com` for 20 GPS systems and 20 TV sets, respectively. The **VEG** and **SFR** datasets include the complete review corpora from `Yelp.com` for 20 Las Vegas Hotels and 20 San Francisco restaurants, respectively. The average number of reviews per item for **GPS**, **TVS**, **VEG** and **SFR** was 203.5, 145, 266 and 968, respectively.

6.2 Qualitative Evidence

First, we present qualitative evidence that demonstrate the validity of the spotlight sets produced by `ImportanceSampling`. Figure 1 shows examples for two different items. For each item, we present the set of considered attributes, which are underlined in the reviews of the respective spotlight set. For lack of space, we consider a subset of the complete set of each item's attributes. We also anonymize the reviews for the sake of discretion. The first spotlight set corresponds to an item from the **TVS** dataset, and contain at least one evaluation for each considered attribute. The second spotlight set follows the variation we discussed at the end of Section 4. In this case, we want to include at least one positive and one negative opinion for each of the item's attributes. In both examples, the spotlight sets produced by our method successfully covers the attributes.

Item 1 (**TVS**), Attributes: { *picture, price, warranty, sound, design, menu*}:

"...Of all the LCD Tvs the ∗ ∗ ∗ overall seemed to have a brighter picture, has 120Hz, 2 year warranty, reasonably priced and..."

"...The ∗ ∗ ∗ delivers outstanding picture quality and sound..."

"...Intuitive menu, easy to plug and play with most any hook up and source... The design of the TV is stunning, beautiful work all around..."

Item 2 (**SFR**), Attributes: {*food, price, staff (service), restrooms (bathrooms)* }

"...The food is delicious, prices are fair, venue is nice, staff is friendly, restrooms are clean..."

"...BAD SERVICE, WORSE ATTITUDES, AND EXTREMELY HIGH PRICES ..."

"...The food was substandard, unfortunately..."

"...the only drawback were the bathrooms..."

Fig. 1. Examples of spotlight sets

6.3 Evaluation of `ImportanceSampling` on the Spotlight-Shuffling Task

Next, we compare our `ImportanceSampling` algorithm against three alternative baselines for spotlight set selection. The considered approaches are evaluated in terms of how well they can balance the compactness of the produced spotlight sets and the visibility they offer to the reviews in the corpus. **(1)** `GreedySampling` begins by selecting a review from the corpus uniformly at random. It then greedily appends reviews, picking the review that covers the most new attributes at every step, until all the attributes at covered. The random choice in the first step is required to introduce variety in the spotlight sequence. The greedy choice at every step is also diversified by randomly picking among all the reviews that maximize the number of new attributes. **(2)** `RandomSampling` populates the spotlight set by picking reviews from the corpus uniformly at random, until all the attributes are covered. **(3)** `HelpSampling` works in a similar manner, except that the probability of choosing a review is proportional to the number of (user-submitted) helpfulness votes it has accumulated.

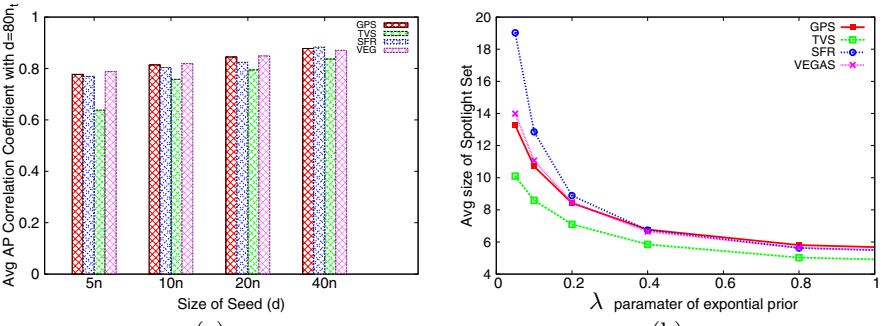
First, we pick the item with the most reviews from each of the four datasets (the results for the remaining items were similar and are omitted for lack of space). We then use each approach to sample 1000 spotlight sets for each item. To account for compactness, we allow for a maximum of 10 reviews per spotlight set. This is also the number of reviews that can fit in a typical webpage `Amazon.com` and is thus in tune with the limited attention span of the average user. If an approach reaches the bound without covering all the attributes, it stops and returns the incomplete set of reviews. To account for this, we report for each approach the percentage of reported sets that were incomplete.

Table 1. Evaluation on the spotlight-set shuffling task

	0	[1-20)	[20-40)	[40-60)	[60-80)	80 ≤	Inc. %
				TVS			
`ImportanceSampling`	0.0	0.67	0.17	0.06	0.05	0.05	8%
`GreedySampling`	0.78	0.14	0.03	0.03	0.0	0.02	0%
`RandomSampling`	0.0	0.0	0.32	0.68	0.0	0.0	90%
`HelpSampling`	0.41	0.19	0.18	0.09	0.02	0.1	48%
				GPS			
`ImportanceSampling`	0.0	0.81	0.08	0.02	0.05	0.04	11%
`GreedySampling`	0.88	0.03	0.03	0.02	0.01	0.04	0%
`RandomSampling`	0.0	0.02	0.92	0.06	0.0	0.0	98%
`HelpSampling`	0.47	0.31	0.09	0.04	0.02	0.07	79%
				SFR			
`ImportanceSampling`	0.15	0.82	0.01	0.02	0.0	0.0	16%
`GreedySampling`	0.93	0.04	0.01	0.0	0.0	0.02	0%
`RandomSampling`	0.0	1.0	0.0	0.0	0.0	0.0	100%
`HelpSampling`	0.5	0.37	0.08	0.02	0.01	0.01	100%
				VEG			
`ImportanceSampling`	0.01	0.81	0.1	0.04	0.02	0.02	14%
`GreedySampling`	0.67	0.27	0.02	0.02	0.0	0.02	0%
`RandomSampling`	0.0	0.09	0.91	0.0	0.0	0.0	99%
`HelpSampling`	0.43	0.15	0.22	0.1	0.05	0.05	97%

The results are shown in Table 1. The second column shows the percentage of reviews that were not included in *any* spotlight set. Columns 2-6 show the number of reviews that were included a number of times that falls within the corresponding intervals (i.e. between 20 and 40 times for column 2). The intervals in Table 1 are chosen based on a step of 20. The prevalence of our method is not affected by this choice. The last column shows the percentage of the reported sets that were incomplete (i.e., they did not cover all the attributes).

The first observation is that `RandomSampling` and `HelpSampling` consistently report high percentages of incomplete spotlight sets, reaching up to 100%. This is due to the fact that they do no consider the complementarity among reviews. As a result, they can require an arbitrarily large number of reviews to be included in order for all the atttributes to be covered. On the other hand, `GreedySampling` reports no incomplete spotlight sets, since it ensures complete coverage with a minimal set of reviews. However, this approach fails to fairly distribute visibility privileges to the reviews. In fact, a consistently high percentage of the reviews were not included in any spotsets. The greedy nature of the algorithm forces it to focus on a small subset of the review population, while completely overlooking the majority. Finally, our `ImportanceSampling` method successfully balances a consistently low percentage of incomplete spotlight sets and a fair distribution of visibility; the percentages of completely neglected reviews (column 1) was



(a) (b)

Fig. 2. Figure 2(a): Stability of `ImportanceSampling` with respect to seed size. Figure 2(b): y-axis: Size of spotlight set, x-axis: value of the parameter λ.

consistently low, going down to zero for two of the items. In general, even though some reviews were rightly sampled more often than others by our approach, almost all the reviews were given a fair chance to be in the spotlight.

6.4 The Effect of the Seed of Minimal Covers on `ImportanceSampling`

As discussed in Section 4, we compute only a subset of the complete set of minimal covers required by `ImportanceSampling`. We call such subset a *seed* and denote it by $\mathcal{M}$. Next, we evaluate the effect of $|\mathcal{M}|$ on the algorithm's results. We used the version of the algorithm given in Section 4.4, with $\lambda = 0.5$. The results for different values of λ were similar and are omitted for lack of space.

Given the review corpus R on an item, we use the method discussed in Section 4 to obtain a seed $\mathcal{M}$ of size d. We then use `ImportanceSampling` to sample spotlight sets from R, until convergence. The standard principles of `ImportanceSampling` are applied to determine convergence [15]. Let count(r) be the number of times a review r was included in a sampled spotlight set. We then rank the reviews by their respective counts. The process is repeated to obtain a new ranking for different values of $d = |\mathcal{M}|$. Figure 2(a) shows the AP correlation coefficient [19], computed by comparing the rankings obtained for $d \in \{n, 5n, 10n, 20n, 40n\}$, with the ranking given for $d = 80n$. AP takes values in $[-1, 1]$ and is a variant of Kendall τ used to compare two rankings by giving more importance to the top of the ranks. Figure 2(a) shows the AP values (y-axis) obtained for different values of d (x-axis). The results clearly show that a seed of size linear to the number of reviews is sufficient to approximate the counts. The AP values steadily increase with d, until, for $d = 40n$, they reach a near-perfect value of 0.9.

6.5 Compactness Evaluation

Here, we evaluate the spotlight sets produced by the `ImportanceSampling` with respect to their compactness. We sample spotlight sets using the pseudocode

given in Algorithm 1, using a seed $\mathcal{M}$ of size $40|R| = 40n$. We repeat the process for different values of $\lambda \in [0.05, 0.2, 0.4, 0.8, 1.6]$, the parameter of the exponential prior. Figure 2(b) shows the average cardinality of the sampled spotlight sets, taken over all items in each dataset (y-axis) as a function of the λ values (x-axis). As expected, increasing the value of λ leads to smaller spotlight sets. We observe that setting $\lambda \in [0.1, 0.2]$ consistently produces spotlight sets of size around 10. This is also the number of reviews that can fit in a typical webpage of `Amazon.com` and is thus in tune with the limited attention span of the average user. In any case, the λ parameter is an intuitive way to tune the size of the presented spotlight sets according to one's own specifications.

6.6 Evaluating the Attribute-Recommendation System

Next, we evaluate our attribute-recommendation system, presented in Section 5. We first pick the item with the most reviews from each of the four datasets. For each item, we randomly pick 100 reviews from its corpus and extend each review r with the set of attributes Q suggested by our method. We then record the ratio of the review's *expected dominance* after the extension, over the number of reviews that r initially dominates (i.e. $\frac{\text{ED}(r \cup Q)}{D(r)+1}$).

The results are shown in Table 2. The first column shows the average ratio over all 100 reviews picked per item. The second column shows the respective standard deviation. The third and fourth columns show the same quantities, calculated only over the subset of the reviews that benefited by the extension. The fifth column shows the maximum ratio observed for each item, and the sixth column shows the percentage of reviews that benefited from the extension. For smoothing purposes, all the average and standard-deviation computations were done after removing the top and bottom 5% of reviews, with respect to the ratio.

Table 2. Results of the Attribute-Recommendation System

	Avg	StDev	Avg*	StDev*	Max	Improved (%)
TVS	1.5	1	2.1	1.2	21	50%
GPS	1.7	1.5	2.5	1.9	8.8	58%
VEG	1.3	0.7	2.2	0.9	12.2	34%
SFR	1.8	1.2	2.6	1.1	20.1	56%

The results show that a significant portion of the reviews consistently benefited from the extension. The average improvement ratio was between 1.3 and 1.8 for all reviews, rising to 2.1 and 2.6 respectively when considering only the benefited reviews. Further, the observed maximum ratios show that an appropriate extension can have a tremendous effect on the dominance of a review. The effects of the recommendation system depend largely on the computation of the $Pr(a|r)$ probabilities (see Section 5); higher probabilities increase the expected dominance (see Equation (1)) and thus make extensions more beneficial.

7 Conclusion

In this paper, we presented a novel review-management system that considers the satisfaction of the customers, as well as the reviewers. We showed how informative and compact sets of reviews can be sampled from a corpus in way that takes into consideration the contribution and quality of each review. Further, we proposed an attribute-recommendation system that can help reviewers improve their reviews in order to gain visibility in the system. We concluded the paper with a thorough experimental evaluation on real review data. Overall, our framework considers both users and reviewers and try to optimize both the user satisfaction and utilization of reviewer expertise while motivating reviewers to submit high-quality content.

Acknowledgments. This research was supported in part by NSF award #1017529 and a gift from Microsoft. We also thank Aris Gionis for useful discussions.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. SIGMOD Rec. 22, 207–216 (1993)
2. Bailey, J., Manoukian, T., Ramamohanarao, K.: A fast algorithm for computing hypergraph transversals and its application in mining emerging patterns. In: ICDM, pp. 485–488 (2003)
3. Eiter, T., Gottlob, G.: Identifying the minimal transversals of a hypergraph and related problems. SIAM J. Comput. 24, 1278–1304 (1995)
4. Ghani, R., Probst, K., Liu, Y., Krema, M., Fano, A.: Text mining for product attribute extraction. SIGKDD Explorations Newsletter (2006)
5. Hébert, C., Bretto, A., Crémilleux, B.: A data mining formalization to improve hypergraph minimal transversal computation. Fundam. Inf. 80, 415–433 (2007)
6. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: SIGKDD (2004)
7. Hu, M., Liu, B.: Mining opinion features in customer reviews. In: AAAI (2004)
8. Jindal, N., Liu, B.: Opinion spam and analysis. In: WSDM 2008 (2008)
9. Kincaid, J., Fishburne, R., Rogers, R., Chissom, B.: Derivation of new readability formulas for navy enlisted personnel. In: Research Branch Report 8-75. Naval Technical Training, Millington (1975)
10. Ku, L.-W., Liang, Y.-T., Chen, H.-H.: Opinion extraction, summarization and tracking in news and blog corpora. In: AAAI-CAAW (2006)
11. Lappas, T., Gunopulos, D.: Efficient confident search in large review corpora. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6322, pp. 195–210. Springer, Heidelberg (2010)
12. Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., Zhou, M.: Low-quality product review detection in opinion summarization. In: EMNLP-CoNLL (2007)
13. Lu, Y., Tsaparas, P., Ntoulas, A., Polanyi, L.: Exploiting social context for review quality prediction. In: WWW 2010 (2010)
14. Min Kim, S., Pantel, P., Chklovski, T., Pennacchiotti, M.: Automatically assessing review helpfulness. In: EMNLP 2006 (2006)

15. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)
16. Popescu, A.-M., Etzioni, O.: Extracting product features and opinions from reviews. In: HLT 2005 (2005)
17. Riloff, E., Patwardhan, S., Wiebe, J.: Feature subsumption for opinion analysis. In: EMNLP (2006)
18. Li, M.X., Tan, C.H., Wei, K.K., Wang, K.L.: Where to place product review? an information search process perspective. In: ICIS (2010)
19. Yilmaz, E., Aslam, J.A., Robertson, S.: A new rank correlation coefficient for information retrieval. In: SIGIR, pp. 587–594 (2008)
20. Zhang, Z., Varadarajan, B.: Utility scoring of product reviews. In: CIKM (2006)
21. Zhuang, L., Jing, F., Zhu, X., Zhang, L.: Movie review mining and summarization. In: CIKM (2006)
22. Tsaparas, P., Ntoulas, A., Terzi.: Constructing a comprehensive set of reviews. In: SIGKDD (2011)

Focused Multi-task Learning Using Gaussian Processes

Gayle Leen[1,2,⋆], Jaakko Peltonen[1,2], and Samuel Kaski[1,2,3]

[1] Aalto University School of Science,
Department of Information and Computer Science
[2] Helsinki Institute of Information Technology HIIT
[3] University of Helsinki, Department of Computer Science
gayle.leen@decode.is, jaakko.peltonen@tkk.fi, samuel.kaski@aalto.fi

Abstract. Given a learning task for a data set, learning it together with related tasks (data sets) can improve performance. Gaussian process models have been applied to such multi-task learning scenarios, based on joint priors for functions underlying the tasks. In previous Gaussian process approaches, all tasks have been assumed to be of equal importance, whereas in transfer learning the goal is *asymmetric*: to enhance performance on a target task given all other tasks. In both settings, transfer learning and joint modelling, *negative transfer* is a key problem: performance may actually decrease if the tasks are not related closely enough. In this paper, we propose a Gaussian process model for the asymmetric setting, which learns to "explain away" non-related variation in the additional tasks, in order to focus on improving performance on the target task. In experiments, our model improves performance compared to single-task learning, symmetric multi-task learning using hierarchical Dirichlet processes, and transfer learning based on predictive structure learning.

Keywords: Gaussian processes, multi-task learning, asymmetric setting, negative transfer.

1 Introduction

Analysis of brain signals is a prime example of data analysis tasks which could benefit from successful transfer learning. Functional neuroimaging studies typically suffer from the "small n, large p" problem: the number of subjects n is small but the dimensionality of the data p for each subject, obtained by methods such as functional Magnetic Resonance Imaging (fMRI) is huge. In patient studies of a brain disorder, there are practical limitations on how many patients can be accessed and measured, and in experimental neuroscience the problem is that the larger the number of replications and variants needed, the less new neuroscience can be done. Moreover, when generalizing across subjects, the brain physiology and function are sufficiently similar that different brains can be matched, but the matching is only approximate. We study a classification task in which the goal

⋆ Now at deCODE Genetics, Reykjavik.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 310–325, 2011.

is to predict the stimulus given brain measurements of a certain user, utilizing the measurements of other users on the same and different stimuli.

The task is more general, however. It has been shown that transferring knowledge between several potentially related learning tasks has improved performance. This scenario, termed multi-task learning [6] or transfer learning [14], has gained considerable attention in the machine learning community in recent years (see [11] for a recent review). Sharing statistical strength between tasks can potentially compensate for having very few samples in the desired learning task, and can make the inference more robust to noise.

1.1 Symmetric and Asymmetric Multi-task Learning

Transfer of knowledge between different tasks is useful only when the tasks are related; if tasks are unrelated, *negative transfer* can occur, meaning that the transfer distorts the model learned for a target task rather than providing additional statistical strength. Therefore, a crucial part of multi-task learning algorithms lies in the modelling of task relatedness, through the specification and the learning of the dependency structure between tasks.

In general, existing multi-task learning approaches use a symmetric dependency structure between tasks. This type of set-up, which we term *symmetric multi-task learning*, assumes that all tasks are of equal importance. The set of related tasks is learned jointly, with the aim of improving over learning the tasks separately (the *no transfer* case), averaged over all tasks.

However, a common learning scenario is to learn a specific task (*primary task*), while incorporating knowledge learned through other similar tasks (*secondary tasks*). For instance, in the neuroscience scenario mentioned earlier, we are interested in learning about a specific patient's response to a stimulus, but we can transfer information from other patients' responses to related stimuli to improve learning. This asymmetric case, or *transfer learning*, requires the assumption of an asymmetric dependency structure between tasks. Existing approaches include reweighting-based methods [16,3,4] or learning of shared feature spaces. An alternative has been to, in effect, use a symmetric multi-task learning method in an asymmetric mode, by using the model learned from auxiliary tasks as a prior for the target task [9,12,17].

Inspired by the Gaussian process (GP) models used earlier for symmetric multi-task learning, we propose a novel and simple dependency structure for asymmetric multi-task learning using GPs. This focuses on learning a target task and learns to avoid negative transfer; this can be done conveniently in the GP formulation, by adding task-specific processes which "explain away" irrelevant properties. At the same time, flexibility of the GP framework is preserved.

2 Dependency Structure in Multi-task Learning with Gaussian Processes

Supervised learning tasks such as classification and regression can be viewed as function approximation problems given the task inputs and targets; accordingly,

multi-task learning can be viewed as learning multiple, related functions. The Gaussian process (GP) framework provides a principled and flexible approach for constructing priors over functions. The GP framework has subsequently been applied successfully to multi-task learning problems [20,5,1]. A crucial element of these models is the way in which the dependency structure between the multiple functions is encoded through the construction of the covariance function. However, current GP approaches do not address the problem of asymmetric multi-task learning, and only consider symmetric dependency structures, which we review in the following subsection.

2.1 Symmetric Dependency Structure

Suppose that there are N distinct inputs, $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top$, and M tasks, such that $y_{t,i}$ is the target for input i in task t. We denote the vector of outputs for task t as $\mathbf{y}^t = [y_1^t, ..., y_N^t]^\top$, and the $N \times M$ vector of outputs for all M tasks, as $\mathbf{y} = [\mathbf{y}_1^\top, ..., \mathbf{y}_M^\top]^\top$. In the GP approach to the problem, it is assumed that there is a latent function underlying each task, $f_1, ..., f_M$. Denoting the latent function evaluated at input i for task t as $f_t(\mathbf{x}_i)$, a (zero mean) GP prior is defined over the latent functions, with a covariance function of the form

$$\langle f_t(\mathbf{x}) f_{t'}(\mathbf{x}') \rangle = k^T(t, t') k^x(\mathbf{x}, \mathbf{x}') \tag{1}$$

where k^T is a covariance function over tasks, specifying the intertask similarities, and k^x is a covariance function over inputs. For regression tasks, the observation model is $y_{i,t} \sim \mathcal{N}(f_t(\mathbf{x}_i), \sigma_t^2)$, where σ_t^2 is the noise variance in task t.

In [5], k^T is defined as a 'free-form' covariance function, where $k^T(i, j) = K^T_{i,j}$ indexes a positive semidefinite intertask similarity matrix K^T. Other methods such as [19] have included a parameterised similarity matrix over task descriptor features, but this could be restrictive in modelling similarities between tasks. These types of priors essentially assume that each of the task latent functions is a linear combination of a further set of latent functions, known as intrinsic correlation models in the geostatistics field (see e.g. [15]). This idea was further generalised in [1] to generating the task latent functions by convolving a further set of latent functions with smoothing kernel functions.

2.2 Predictive Mean for Symmetric Multi-task GP

The predictive mean on a new data point $\mathbf{x}_*$ in task j, for the multi-task GP formulation of [5], is given by

$$\overline{f}_j(\mathbf{x}_*) = (\mathbf{k}_j^T \otimes \mathbf{k}_*^x)^\top \Sigma^{-1} \mathbf{y} \quad \text{where} \quad \Sigma = K^T \otimes k^x(\mathbf{X}, \mathbf{X}) + D \otimes \mathbf{I} \tag{2}$$

where $\mathbf{k}_j^T$ is the jth column of task similarity matrix K^T, $\otimes$ is the Kronecker product, $\mathbf{k}_*^x = [k(\mathbf{x}_*, \mathbf{x}_1), ..., k(\mathbf{x}_*, \mathbf{x}_N)]^\top$ is the vector of covariances between the test input $\mathbf{x}_*$ and the training inputs. The $k^x(\mathbf{X}, \mathbf{X})$ is the matrix of covariance function values between all training input points, and D is an $M \times M$ diagonal matrix where the (j, j)th element is σ_i^2.

To gain intuition into the form of the predictive mean, let us define the $M \times N$ vector $\mathbf{w} = \Sigma^{-1}\mathbf{y}$, and divide it into M blocks of N elements: $\mathbf{w} = [\mathbf{w}_1^\top, ..., \mathbf{w}_M^\top]^\top$. We can then rewrite (2) as

$$\overline{f}_j(\mathbf{x}_*) = \sum_{m=1}^{M} K^T_{m,j}(\mathbf{k}^x_*)^\top \mathbf{w}_m = \sum_{m=1}^{M} K^T_{m,j}\mu^m_* \tag{3}$$

where $\mu^m_* = (\mathbf{k}^x_*)^\top \mathbf{w}_m$ can be interpreted as the posterior mean of the latent function at $\mathbf{x}_*$ for task m, thus (2) is a weighted sum of posterior means for all tasks, and the weights $\{K^T_{m,i}\}_{m=1}^M$ are covariances between task j and all tasks.

Since K^T is positive semidefinite, the sharing of information between tasks is naturally symmetric, and all tasks are treated equally. However, we are interested in an asymmetric setup, where we learn a primary task together with several secondary tasks. Rather than modelling the relationships between secondary tasks, we want to focus on the aspects relevant to learning the primary task.

2.3 Asymmetric Dependency Structure

In the previous symmetric learning problem, the tasks were modelled as conditionally independent on a set of M (i.i.d.) underlying functions, which capture the shared structure between all tasks. In this section, we derive an asymmetric version of a GP framework for multi-task learning, by constraining the secondary tasks to be conditionally independent given the primary task, such that the shared structure between all secondary tasks is due to the primary task.

Similarly to the previous notation, let us denote the inputs to each task as $\mathbf{X}$. Suppose that there is one primary task, with targets $\mathbf{y}^p = [y^p_1, ..., y^p_N]^\top$, with underlying latent function values $\mathbf{f}^p = [f^p(\mathbf{x}_1), ..., f^p(\mathbf{x}_N)]^\top$. Suppose there are $M-1$ secondary tasks, where the targets for the ith secondary task are denoted by $\mathbf{y}^{s_i} = [y^{s_i}_1, ..., y^{s_i}_N]^\top$. The corresponding latent function values are $\mathbf{f}^{s_i} = [f^{s_i}(\mathbf{x}_1), ..., f^{s_i}(\mathbf{x}_N)]^\top$.

We are interested in learning the underlying function f^p for the primary task. Here, potentially related secondary tasks can help to learn f^p; conversely if we know f^p, this could help to learn the functions underlying the secondary tasks $\{f^{s_i}\}$. We can formalise this intuition by examining the GP predictive likelihood on the secondary task function values, after training on the primary task. However, first we need to define a joint prior over the primary and secondary task function values. We start by making the assumption that $\{f^{s_i}\}$ can be decomposed into a 'shared' component (which is shared with the primary task) and a 'specific' component. That is, for the nth input,

$$f^{s_i}(\mathbf{x}_n) = f^{s_i,shared}(\mathbf{x}_n) + f^{s_i,specific}(\mathbf{x}_n)\ . \tag{4}$$

Further we assume that $f^{s_i,shared} = \rho_{s_i} f^p$, that is, the shared component is correlated with the primary task function. This may seem like a restrictive assumption but assuming linear relationships between task functions has been proved to be successful in e.g. [15,5]. Now we can place a shared prior over each $f^{s_i,shared}$ and f^p. The corresponding graphical model is presented in Figure 1.

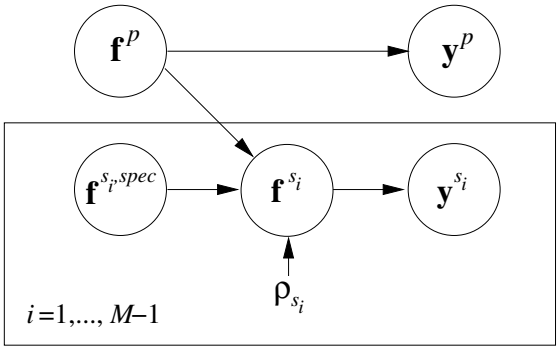


Fig. 1. Graphical model of the focused GP multi-task model, showing the relationship between the function values of the primary and secondary tasks. Parameters of the covariance functions omitted for clarity.

Sharing between Primary and Secondary Task Functions. We place a zero mean Gaussian process prior on f^p, with covariance function k^p, such that the prior on the shared function is also a GP, with covariance function

$$\left\langle f^t(\mathbf{x}) f^{t'}(\mathbf{x}') \right\rangle = k^t(t,t') k^p(\mathbf{x},\mathbf{x}') \quad \text{where} \quad k^t(t,t') = \rho_t \rho_{t'} \tag{5}$$

where ρ_t is the correlation of task t with the primary task, and $\rho_p = 1$, and f^t can denote either the primary task or any of the secondary tasks. Denoting the task functions for the $M-1$ secondary tasks as $\mathbf{f}^s = \left[(\mathbf{f}^{s_1})^\top, ..., (\mathbf{f}^{s_{M-1}})^\top\right]^\top$, the joint distribution over the shared function values is given by

$$p(\mathbf{f}^p, \mathbf{f}^{s,shared}) = \mathcal{GP}\left(0, \begin{bmatrix} \mathbf{K}_{pp} & \mathbf{K}_{sp}^\top \\ \mathbf{K}_{sp} & \mathbf{K}_{ss} \end{bmatrix}\right) \tag{6}$$

where $\mathbf{K}_{pp}$ is the matrix of covariance function values from (5) between the primary task points, $\mathbf{K}_{sp}$ evaluated between secondary and primary, and $\mathbf{K}_{ss}$ between secondary task inputs. Given the primary task function values, we can then derive the predictive distribution on the shared components of the secondary tasks using the standard GP equations:

$$p(\mathbf{f}^{s,shared} \mid \mathbf{f}^p) = \mathcal{GP}\left(\mathbf{K}_{sp}\mathbf{K}_{pp}^{-1}\mathbf{f}^p, \Lambda\right) \tag{7}$$

where Λ is a diagonal matrix whose elements are given by the diagonal of $\mathbf{K}_{ss} - \mathbf{K}_{sp}\mathbf{K}_{pp}^{-1}\mathbf{K}_{sp}^\top$. This approximation allows us to make a reduced rank approximation, and offers a computationally efficient solution to jointly learning the covariance matrix across a large number of input points.

An interpretation of equation (7) is the posterior distribution of f^p (the primary task function) after observing the primary task function values $\mathbf{f}^p$, evaluated at all the secondary task inputs. This differs slightly from the standard GP predictive equations in that the posterior mean for each secondary task s is

weighted by ρ_s, which models the correlation with the primary task. To illustrate this, for secondary task l, the posterior mean $\bar{f}^{1,shared}$ given $\mathbf{f}^p$:

$$\bar{f}^{1,shared} = \rho_l k^p(\mathbf{X}_l, \mathbf{X}_p) k^p(\mathbf{X}_p, \mathbf{X}_p)^{-1} \mathbf{f}^p = \rho_l \mu_l^p$$

where we have used the notation: $\mathbf{X}_i$ is the set of input points for task i, and μ_l^p is the posterior mean given covariance function k^p and observations $\mathbf{f}^p$, evaluated at $\mathbf{X}_l$. Learning ρ_s during training can help to avoid negative transfer from secondary to primary task.

Explaining Away Secondary Task-Specific Variation. We define the covariance function over $\mathbf{f}^{s,specific}$ to be block diagonal in $[\mathbf{K}_1^{spec}, ..., \mathbf{K}_{M-1}^{spec}]$ with respect to the tasks. These covariance functions have parameters specific to each task: $\mathbf{f}^{s,specific} \sim \mathcal{GP}(0, \mathbf{K}^{spec})$. This creates flexible models for the secondary tasks, which can 'explain away' variation that is specific to a secondary task, and unshared with the primary task. Since the primary task function values are unknown, rather than estimating them directly we integrate over them:

$$p(\mathbf{f}^s) = \mathcal{GP}\left(0, \mathbf{K}_{sp}\mathbf{K}_{pp}^{-1}\mathbf{K}_{sp}^\top + \Lambda + \mathbf{K}^{spec}\right) . \tag{8}$$

Putting everything together, the resulting prior on all the task functions is

$$p(\mathbf{f}^p, \mathbf{f}^s) = \mathcal{GP}\left(0, \begin{bmatrix} \mathbf{K}_{pp} & \mathbf{K}_{sp}^\top \\ \mathbf{K}_{sp} & \mathbf{K}_{sp}\mathbf{K}_{pp}^{-1}\mathbf{K}_{sp}^\top + \Lambda + \mathbf{K}^{spec} \end{bmatrix}\right) \tag{9}$$

2.4 Hyperparameter Learning

We can learn the hyperparameters of our model in (9) by optimising the marginal log likelihood with respect to the hyperparameters of the covariance functions, the task similarity vector $[\rho_{s_1}, ..., \rho_{s_{M-1}}]$, and the parameters of the observation model, given the inputs $\mathbf{x}$ and targets y. For regression, the observation model is $y_{i,t} \sim \mathcal{N}(f_t(\mathbf{x}_i), \sigma_t^2)$, where σ_t^2 is the noise variance in task t, and for classification we use a probit noise model $p(y_{i,t} \mid f_{i,t} = \Phi(y_{i,t}(f_{i,t} + b))$, where Φ is the cumulative distribution function for a standard Gaussian $\mathcal{N}(0, 1)$, and b is a bias parameter. For the binary classification experiments in Section 5.2, we make an approximation to the model likelihood using Expectation Propagation [10].

3 Related Work and Discussion

In our focused multi-task GP, the pseudo-input locations are fixed as the inputs to the primary task, such that they can explain the shared variation between the primary and secondary tasks, and also between the secondary tasks. The sparse GP method in [13] bears similarities to our model. This parameterises the covariance function of a GP by learning a set of pseudo-input locations. In that model, the pseudo-inputs summarise the variation of the data through assuming that the function values are conditionally independent given the pseudo-inputs.

Recently there has been interest in asymmetrical GP multi-task learning [7], where generalisation errors for the multi-task GP of [5] were derived for an asymmetrical multi-task case, with one primary and one secondary task. However, this work did not derive a new model for asymmetric multi-task learning, and focused on analysing the symmetric model.

The asymmetric dependency structure that we have presented uses a simple idea to bias the model to learning the underlying function for the primary task, by decomposing the underlying task functions for the secondary tasks as 'shared' and 'specific' components. The shared components are from a joint GP prior with the primary task function. These are conditioned on the primary task function values (7) such that this biases the shared variation between tasks to be due to the primary task function, and a task specific weight, which is learned during training. We additionally assume that the each of the secondary task functions can also be explained by a process specific to it, by defining a block diagonal covariance structure over the secondary tasks. This allows the model to 'explain away' secondary task specific variation and focus the model on learning the primary task.

In this first paper we make the simplifying assumption that the task of interest is entirely composed of the shared function, and that there are no other strong shared functions between other tasks. This model already proves useful in a challenging fMRI task, demonstrating that the idea of asymmetric modelling with explaining-away yields useful results, and it can be extended to more general asymmetric modelling in later stages.

In brief, if there is reason to suspect detrimental shared variation between other tasks, one can add additional GP functions which is shared between other tasks but not with the primary task. The overall model can then learn which shared function is a better explanation. As the number of tasks increases, the number of possible sharing configurations increases (shared functions between 2,3,...,M tasks) and the complexity of the model quickly increases. This will be studied in further work.

4 Examining the Generalisation Error for Asymmetric and Symmetric Models

To examine the effect of the processes that are specific to a secondary task, we look at the generalisation error on the primary task for the asymmetric two tasks case in a similar manner to [7]. We investigate the influence of ρ, the degree of "relatedness" between the two tasks. Suppose that we have training inputs $\mathbf{X}_P$ for the primary task, and $\mathbf{X}_S$ for the secondary task. The covariance matrices C_{sym} and C_{asym}, for the symmetric and asymmetric cases respectively, of the noisy training data are given by:

Symmetric case

$$C_{\text{sym}}(\rho) = K^{\text{sym}}(\rho) + \sigma_n^2 \mathbf{I} \quad \text{where} \quad K^{\text{sym}}(\rho) = \begin{pmatrix} K^p_{PP} & \rho K^p_{PS} \\ \rho K^p_{SP} & K^p_{SS} \end{pmatrix} \tag{10}$$

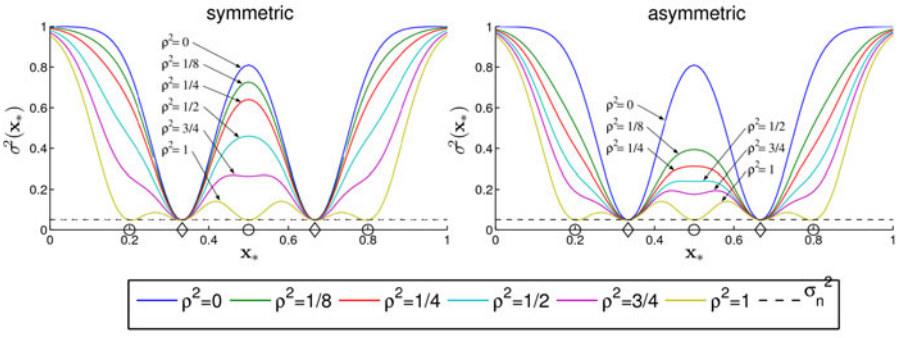


Fig. 2. The posterior variances for the test locations $\mathbf{x}_* \in [0, 1]$ given training points from the primary task ($\mathbf{X}_P = [1/3 \;\; 2/3]$, plotted as $\diamond$) and secondary task ($\mathbf{X}_S = [1/5 \;\; 1/2 \;\; 4/5]$, plotted as $\circ$) for the symmetric case (top) and the asymmetric case (bottom). Each plot uses corresponding values of ρ^2 (see legend).

Asymmetric case

$$C_{\text{asym}}(\rho) = K^{\text{asym}}(\rho) + \sigma_n^2 \mathbf{I}$$
$$\text{where} \;\; K^{\text{asym}}(\rho) = \begin{pmatrix} K^p_{PP} & \rho K^p_{PS} \\ \rho K^p_{SP} & \rho^2 K^p_{SS} + (1-\rho^2) K^s_{SS} \end{pmatrix} \tag{11}$$

where we have used the notation K^p_{AB} to denote the matrix of covariance values, due to k^p, evaluated between $\mathbf{X}_A$ and $\mathbf{X}_B$. For the asymmetric case, the covariance matrix for the secondary task comes from the 'shared' covariance function k^p with the primary task, and a 'specific' covariance function k^s. The relationship between the primary and secondary tasks due to the ρ's comes directly from (1) and (5) for the symmetric and asymmetric cases respectively.

4.1 Generalisation Error for a Test Point $\mathbf{x}_*$

If the GP prior is correctly specified, then the posterior variance for a new test point $\mathbf{x}_*$ for the primary task (due to the noise free f^p) is also the generalisation error for $\mathbf{x}_*$. The posterior variance at $\mathbf{x}_*$ for the primary task is:

$$\textbf{Symmetric case:} \quad \sigma^2_{\text{sym}}(\mathbf{x}_*, \rho) = k_{**} - \mathbf{k}_*^\top C_{\text{sym}}(\rho)^{-1} \mathbf{k}_* \tag{12}$$
$$\textbf{Asymmetric case:} \quad \sigma^2_{\text{asym}}(\mathbf{x}_*, \rho) = k_{**} - \mathbf{k}_*^\top C_{\text{asym}}(\rho)^{-1} \mathbf{k}_* \tag{13}$$

where k_{**} is the prior variance at $\mathbf{x}_*$, $k^p(\mathbf{x}_*, \mathbf{x}_*)$, and $\mathbf{k}_*^\top = (k^p(\mathbf{x}_*, \mathbf{X}_p) \; \rho k^p(\mathbf{x}_*, \mathbf{X}_s))$ We note that the target values y do not affect the posterior variance at the test locations, and have omitted the dependence on $\mathbf{X}_P$, $\mathbf{X}_S$ and σ_n^2 in the notation for $\sigma^2_{\text{sym}}(\mathbf{x}_*, \rho), \sigma^2_{\text{asym}}(\mathbf{x}_*, \rho)$ for clarity.

To illustrate the difference between the symmetric and asymmetric cases, we plot the posterior variances as a function of $\mathbf{x}_*$ in Figure 2, given two observations for the primary task, and three observations of the secondary task (see figure for more details). Following the setup in [7], we use a squared exponential covariance

function with lengthscale 0.11 for k^p, noise variance $\sigma_n^2 = 0.05$, and, for the asymmetric setup, a squared exponential covariance function with lengthscale 1 for k^s.

Each plot contains 6 curves corresponding to $\rho^2 = [0, 1/8, 1/4, 1/2, 3/4, 1]$, and the dashed line shows the prior noise variance. The training points from the primary task ($\diamond$) create a depression that reaches the prior noise variance for all the curves. However, the depression created by the training points for the secondary task ($\circ$) depends on ρ. For the single task learning case ($\rho = 0$), there is no knowledge transferred from the secondary task. As ρ increases, the generalisation error at the secondary task test points decreases. For the intermediate ρ^2 values (i.e. not 0 or 1 (full correlation)), our asymmetric model gives a smaller posterior variance than the symmetric model at secondary task locations, and therefore suggests better generalisation error.

4.2 Intuition about the Generalisation Errors

Given the illustrative example in the previous section, we sketch the relationship between the generalisation errors for the primary and secondary tasks:

$$\sigma^2_{\text{asym}}(\mathbf{x}_*, \rho) \leq \sigma^2_{\text{sym}}(\mathbf{x}_*, \rho) \tag{14}$$

We show this by considering the covariance matrix at the secondary task points, conditioned on the primary task points. This represents the residual uncertainty about the secondary task points, given that we know the primary task points. Denoting this quantity as $A(\rho)$:

$$A(\rho)_{\text{sym}} = K^p_{SS} + \sigma_n^2\mathbf{I} - \rho^2 K^p_{SP}(K^p_{PP} + \sigma_n^2\mathbf{I})^{-1}K^p_{PS} \tag{15}$$

$$A(\rho)_{\text{asym}} = \rho^2 K^p_{SS} + (1-\rho^2)K^s_{SS} + \sigma_n^2\mathbf{I} - \rho^2 K^p_{SP}(K^p_{PP} + \sigma_n^2\mathbf{I})^{-1}K^p_{PS} \tag{16}$$

If $A(\rho)_{\text{asym}} \preceq A(\rho)_{\text{sym}}$ then:

$$\begin{aligned}
A(\rho)^{-1}_{\text{asym}} &\succeq A(\rho)^{-1}_{\text{sym}} \\
\mathbf{v}(\rho)^\top A(\rho)^{-1}_{\text{asym}}\mathbf{v}(\rho) &\geq \mathbf{v}(\rho)^\top A(\rho)^{-1}_{\text{sym}}\mathbf{v}(\rho) \\
k_{**} - k^p(\mathbf{x}_*, \mathbf{X}_P)(K^p_{PP} + \sigma_n^2\mathbf{I})^{-1}k^p(\mathbf{x}_*, \mathbf{X}_P) - \mathbf{v}(\rho)^\top A(\rho)^{-1}_{\text{asym}}\mathbf{v}(\rho) & \\
\leq k_{**} - k^p(\mathbf{x}_*, \mathbf{X}_P)(K^p_{PP} + \sigma_n^2\mathbf{I})^{-1}k^p(\mathbf{x}_*, \mathbf{X}_P) - \mathbf{v}(\rho)^\top A(\rho)^{-1}_{\text{sym}}\mathbf{v}(\rho) & \\
\sigma^2_{\text{asym}}(\mathbf{x}_*, \rho) &\leq \sigma^2_{\text{sym}}(\mathbf{x}_*, \rho)
\end{aligned} \tag{17}$$

where we have used the Banachiewicz inversion formula to evaluate the matrix inversions in (12) and (13), and we have defined $\mathbf{v}(\rho) = \rho(k^p(\mathbf{X}_S, \mathbf{x}_*) - K^p_{SP}(K^p_{PP} + \sigma_n^2\mathbf{I})^{-1}k^p(\mathbf{X}_P, \mathbf{x}_*))$

The asymmetric model has more flexibility than the symmetric model in the modelling of the secondary task, since it uses both f^p and f^s, rather than just f^p. We expect that $A(\rho)$ for the asymmetric version would be smaller than for the symmetric since the additional flexibility should allow more accurate modelling of the covariances between the secondary task points, and hence the asymmetric generalisation error should be smaller than the symmetric.

5 Experiments

In this section, we demonstrate the performance of the focused multi-task GP model on a synthetic regression problem, and compare it with alternative models on an asymmetric multi-task classification problem on fMRI data. In all experiments, we use squared exponential covariance functions with automatic relevance determination (ARD) prior: $k(\mathbf{x}, \mathbf{x}') = \sigma_s^2 \exp(-\frac{1}{2}\sum_d (\mathbf{x}_d - \mathbf{x}'_d)^2 / l_d^2)$, where σ_s^2 is the overall scale and l_d is the lengthscale for the dth input dimension, initialized to 1. This prior is used for both primary and secondary task functions.

5.1 Synthetic Data

Synthetic data are generated as follows (see Fig. 3. All the functions are functions of the same input $\mathbf{x}$, 100 samples evenly spaced on the interval [-5, 5]. The primary task function is generated from $\mathbf{f}_p \sim \mathcal{GP}(0, \mathbf{K}_p)$, where the kernel function is squared exponential with length scale 1. The secondary task functions are generated according to $f_s^m \sim \mathcal{GP}(\alpha_m \mathbf{f}_p, \beta_m \mathbf{K}_s^m)$. Each specific kernel function is squared exponential with lengthscale 1, and α_m is drawn at random

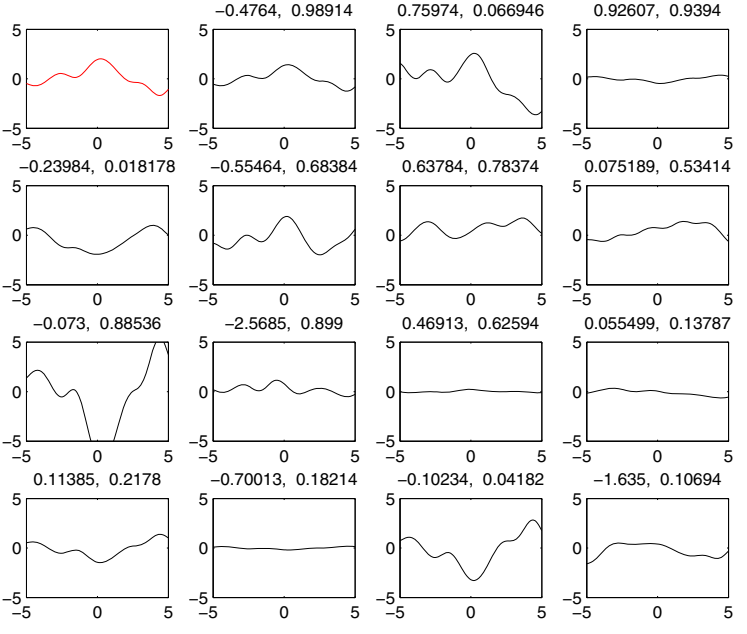
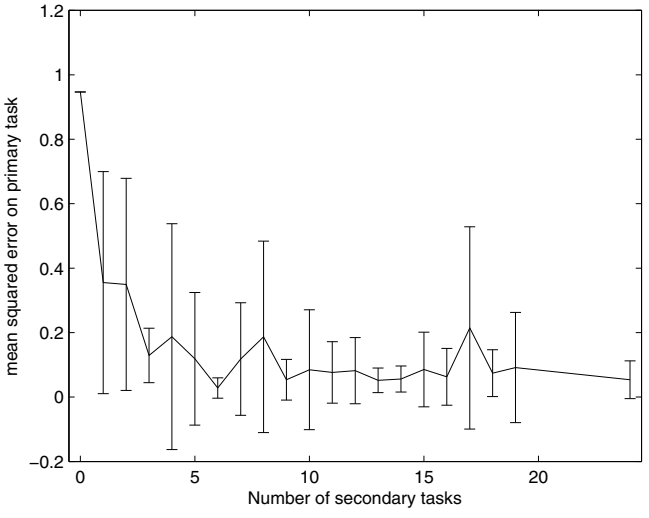
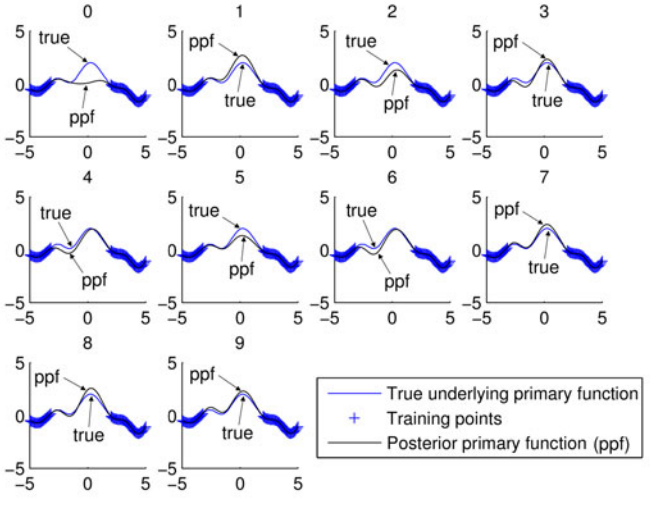


Fig. 3. Synthetic data experiment: experiment setup. We show the functions underlying the generated data: the primary task function (top left, red) and 15 examples of secondary task functions (black). The weights of the shared and specific functions for the secondary tasks are given above each plot.



(a)



(b)

Fig. 4. Synthetic data experiment: results of learning with the proposed asymmetric multi-task Gaussian process model. (a) Mean squared error on the primary task test set, over 10 runs, for different numbers of secondary tasks, error bars represent ± 1 s.d. (b) Posterior distribution over the primary task function for different numbers of secondary tasks (given above each plot).

from $\mathcal{N}(0,1)$, β_m at random from $[0,1]$. We assume a Gaussian observation noise model, since this is a regression problem.

We remove 50 samples from the primary task (see Fig. 4b), and use them as test data. We train the model with different numbers of secondary tasks, ranging from 0 (single task learning) to 24. We repeat the procedure 10 times, randomly drawing the secondary task functions for each run. Figure 4 (b) shows the mean of the posterior distribution (black) over the primary task function for one of the runs, for different numbers of secondary tasks. We also plot the true underlying primary function (blue line), showing that the model can predict the missing part of the primary task function by transferring information from secondary tasks. Figure 4 (a) shows that the mean squared error on the test set decreases as the number of secondary tasks increases.

5.2 fMRI Data

We evaluate the performance of our model on fMRI data, taken from [8]. Six healthy young adults participated in two identical sessions, in which they received a continuous 8-min sequence comprising of auditory, visual and tactile stimuli in blocks of 6×33s. The stimuli of different senses never overlapped. Whole-head volumes were acquired with a Signa VH/i 3.0 T MRI scanner (General Electric, Milwaukee, WI) using a gradient EPI sequence (TR = 3 s, TE = 32 ms, FOV = 20 cm, flip = 90^o, $64 \times 64 \times 44$ voxels with resolution $3 \times 3 \times 3\text{mm}^3$). In each session, 165 volumes were recorded with the 4 first time points excluded from further analysis. Preprocessing of the fMRI data included realignment, normalization with skull stripping, and smoothing. For additional details on the measurements and applied preprocessing, see [18]. After preprocessing, the dimensionality was reduced to 40 by spatial independent component analysis (ICA) that identified spatial brain activation patterns related to various aspects of the stimuli. For each adult, the resulting data is 161 sets of ICA features (40 dimensional), which can be classified according to one of 6 stimuli ('touch', 'auditory' (tones, history, instruction), 'visual' (faces, hands, buildings)).

We consider the task of predicting whether a subject is reacting to a particular stimulus, 'touch', given the fMRI data. We aim to improve the learning of this primary task by learning it in conjunction with other, related tasks from the other subjects. This can be formulated as 6 one-against-all classification tasks in an asymmetric multi-task setup (see Table 1). For each subject the fMRI measurements were done in two separate sessions; in the experiments we use the first session as training data and the second session as test data.

We compare the focused multi-task learning approach ('focused MT-GP') with four reference models. The first baseline model is single task learning using GP classification ('single task GP'), trained only on the samples of the primary task. The second ('pooled GP') learns a GP classification model from the training examples from all tasks (i.e. treating all data as a single task) For 'pooled GP' we use a sparse approximation when the number of training examples > 300, using 30 pseudo-inputs. We also compare to two state-of-the-art methods, one developed for transfer learning and the other multi-task learning: the predictive structure learning method of [2] ('AZ'), and the symmetric multi-task learning with Dirichlet process priors method ('DP-MT') from [17]. For the 'AZ'

Table 1. Asymmetrical multi-task set up for fMRI data study

Subject	Classification Task
1 (primary)	'touch' against all
2 (secondary)	'touch' against all
3 (secondary)	'touch' against all
4 (secondary)	'touch' against all
5 (secondary)	'auditory' (instruction) against all
6 (secondary)	'visual' (buildings) against all

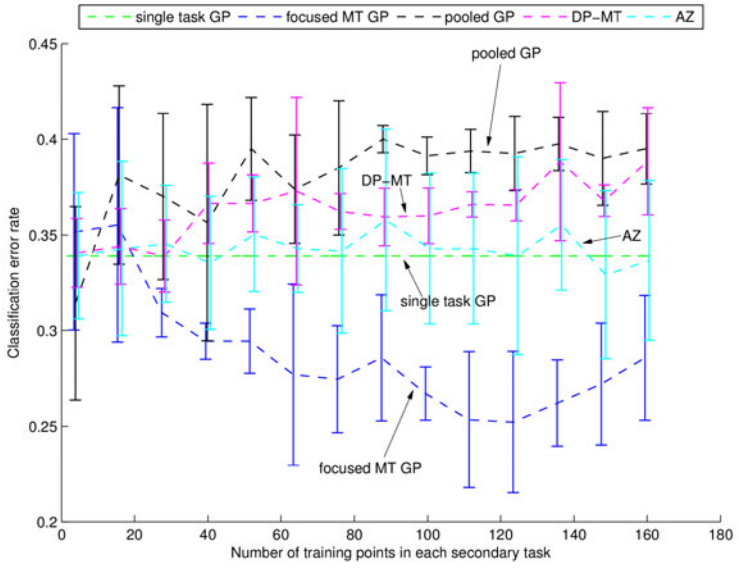
method, the we fix the dimension of the shared predictive structure heuristically to $h = 26$, after performing PCA across all the training samples (primary and secondary) and find the dimension of the subspace that explains 80% of the variance.

We evaluate the methods using a fixed number of training examples in the primary task (64 and 161), while varying the number of training examples in each secondary task (ranging from 4 to 160)), over 5 repetitions. Due to the class imbalance in the data, when randomly picking a subset of secondary training task examples, we ensure that there is at least one positive and one negative example. For the GP-based methods, we also fix the bias parameter $b = \Phi^{-1}(r)$, where r is the ratio of positive samples to negative samples in the training data.

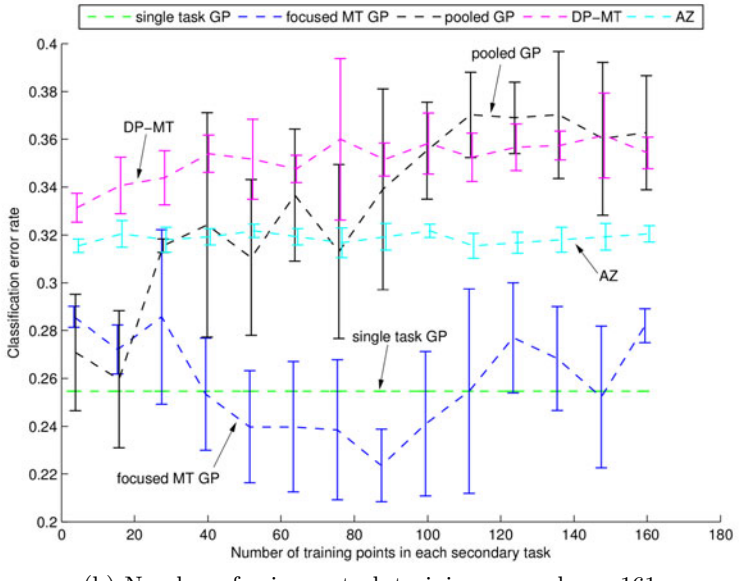
Figure 5 displays the classification error on the test set for the primary task, over different numbers of training examples for the secondary tasks, for 64 training examples in the primary task (a) and 161 (*i.e.*, all available training examples for the primary task) in (b).

Pooling of samples seems to always be a bad choice on this data and, somewhat surprisingly, DP-MT does not work well either. Both work only roughly equally to single-task learning for small numbers of secondary task data and the performance worsens as amount of secondary data increases. Hence it seems that the secondary data here differs from primary data to the extent of causing negative transfer. AZ seems to work better but at most on the same level as single task learning. More work would be needed for model selection, however, which might improve performance.

Focused MT-GP seems able to leverage on the secondary tasks, clearly outperfoming others including single task learning when the amount of data in the primary task is small. Multitask learning is most relevant when the primary task has little data; Focused MT-GP performs well in this scenario. When primary data has more data single task learning improves rapidly, although in Figure 5 Focused MT-GP still outperforms it. Focused MT-GP seems to need more than a few samples in the secondary tasks in order to perform well; the explanation is probably that for this data it is hard to distinguish between useful and negative transfer, and more data is needed to make the choice. Bad performance of pooling and symmetric multi task approaches supports this interpretation.



(a) Number of primary task training examples = 64



(b) Number of primary task training examples = 161

Fig. 5. Classification error on test set for primary task, against number of training examples in each secondary task for different primary task training set size (a: small, b: larger)

6 Conclusion

We derived a multi-task Gaussian process learning method, the 'focused multi-task GP', designed for asymmetrical multi-task learning scenarios, to facilitate improved learning on a primary task through the transfer of relevant information from a set of potentially related secondary tasks. The novel dependency structure was formulated based on the GP predictive distribution over the secondary tasks given the primary task, and constraining the secondary tasks to be conditionally independent. After observing the primary task, the primary task function can be used to predict a part of each secondary task, depending on the degree of task relatedness, which is learned during the optimisation. The model also permits each secondary task to have its own task-specific variation which is unshared with the primary task, and this flexibility should cause the model to focus on modelling the primary task function well. We demonstrated the model on synthetic data and an asymmetrical multi-task learning problem with fMRI data, and showed improved performance over baseline approaches, and a state of the art transfer learning and multi-task learning method.

Acknowledgments. The authors belong to the Adaptive Informatics Research Centre, a national CoE of the Academy of Finland. JP was supported by the Academy of Finland, decision number 123983. This work was also supported in part by the PASCAL2 Network of Excellence, ICT 216886, and by the Tekes Multibio project.

References

1. Alvarez, M., Lawrence, N.D.: Sparse convolved Gaussian processes for multioutput regression. In: Advances in Neural Information Processing Systems, vol. 21, pp. 57–64 (2009)
2. Ando, R.K., Zhang, T.: A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research 6, 1817–1853 (2005)
3. Bickel, S., Bogojeska, J., Lengauer, T., Scheffer, T.: Multi-task learning for HIV therapy screening. In: McCallum, A., Roweis, S. (eds.) Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008), pp. 56–63. Omnipress (2008)
4. Bickel, S., Sawade, C., Scheffer, T.: Transfer learning by distribution matching for targeted advertising. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 21, pp. 145–152 (2009)
5. Bonilla, E.V., Chai, K.M.A., Williams, C.K.I.: Multi-task Gaussian Process Prediction. In: Neural Information Processing Systems (2008)
6. Caruana, R.: Multitask learning. Machine Learning 28, 41–75 (1997)
7. Chai, K.M.A.: Generalization errors and learning curves for regression with multi-task gaussian processes. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems, vol. 22, pp. 279–287 (2009)
8. Malinen, S., Hlushchuk, Y., Hari, R.: Towards natural stimulation in fMRI - issues of data analysis. Neuroimage 35(1), 131–139 (2007)

9. Marx, Z., Rosenstein, M.T., Kaelbling, L.P.: Transfer learning with an ensemble of background tasks. In: Inductive Transfer: 10 Years Later, NIPS 2005 Workshop (2005)
10. Minka, T.: Expectation Propagation for approximative Bayesian inference. In: Breese, J.S., Koller, D. (eds.) Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, pp. 362–369 (2001)
11. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering (in press)
12. Raina, R., Ng, A.Y., Koller, D.: Transfer learning by constructing informative priors. In: Inductive Transfer: 10 Years Later, NIPS 2005 Workshop (2005)
13. Snelson, E., Ghahramani, Z.: Sparse Gaussian Processes using Pseudo-inputs. In: Advances in Neural Information Processing Systems, vol. 18 (2006)
14. Thrun, S.: Is learning the n-th thing any easier than learning the first? In: Advances in Neural Information Processing Systems, vol. 8 (1996)
15. Wackernagel, H.: Cokriging versus kriging in regionalized multivariate data analysis. Geoderma 62, 83–92 (1994)
16. Wu, P., Dietterich, T.G.: Improving SVM accuracy by training on auxiliary data sources. In: Greiner, R., Schuurmans, D. (eds.) Proceedings of the 21st International Conference on Machine Learning (ICML 2004), pp. 871–878. Omnipress, Madison (2004)
17. Xue, Y., Liao, X., Carin, L.: Multi-Task Learning for Classification with Dirichlet Process Priors. Journal of Machine Learning Research 8, 35–63 (2007)
18. Ylipaavalniemi, J., Savia, E., Malinen, S., Hari, R., Vigário, R., Kaski, S.: Dependencies between stimuli and spatially independent fMRI sources: Towards brain correlates of natural stimuli. Neuroimage 48, 176–185 (2009)
19. Yu, K., Chu, W., Yu, S., Tresp, V., Zhao, X.: Stochastic Relational Models for Discriminative Link Prediction. In: Advances in Neural Information Processing Systems, vol. 19 (2007)
20. Yu, K., Tresp, V.: Learning to learn and collaborative filtering. In: Inductive Transfer: 10 Years Later, NIPS 2005 Workshop (2005)

Reinforcement Learning through Global Stochastic Search in N-MDPs

Matteo Leonetti[1], Luca Iocchi[1], and Subramanian Ramamoorthy[2]

[1] Department of Computer and System Sciences,
Sapienza University of Rome,
via Ariosto 25, Rome 00185, Italy
[2] School of Informatics,
University of Edinburgh,
10 Crichton Street, Edinburgh EH8 9AB, United Kingdom

Abstract. Reinforcement Learning (RL) in either fully or partially observable domains usually poses a requirement on the knowledge representation in order to be sound: the underlying stochastic process must be Markovian. In many applications, including those involving interactions between multiple agents (e.g., humans and robots), sources of uncertainty affect rewards and transition dynamics in such a way that a Markovian representation would be computationally very expensive. An alternative formulation of the decision problem involves partially specified behaviors with choice points. While this reduces the complexity of the policy space that must be explored - something that is crucial for realistic autonomous agents that must bound search time - it does render the domain Non-Markovian. In this paper, we present a novel algorithm for reinforcement learning in Non-Markovian domains. Our algorithm, Stochastic Search Monte Carlo, performs a global stochastic search in policy space, shaping the distribution from which the next policy is selected by estimating an upper bound on the value of each action. We experimentally show how, in challenging domains for RL, high-level decisions in Non-Markovian processes can lead to a behavior that is at least as good as the one learned by traditional algorithms, and can be achieved with significantly fewer samples.

Keywords: Reinforcement Learning.

1 Introduction

Reinforcement Learning (RL) in its traditional formulation has been successfully applied to a number of domains specifically devised as test beds, while scaling to large, and more realistic applications is still an issue. RL algorithms, either *flat* or hierarchical, are usually grounded on the model of Markov Decision Processes (MDPs), that represent controllable, fully observable, stochastic domains. When the environment is not observable and not so well understood, however, traditional RL methods are less effective and might not converge. For this reason, the

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 326–340, 2011.

vast majority of work on RL has focused on Markovian domains, and the algorithms for abstracting over the state space, creating a more compact knowledge representation, are designed to ensure the Markov property is maintained.

If the application does not allow the designer to create a reliable description of the state space, such that the representation is Markovian, MDPs are no longer suitable. Partially Observable MDPs overcome part of this limitation: they allow the actual state space to be accessed only through *observations*, but still require the specification of such a space so that an underlying Markovian process exists. Where applicable, POMDPs scale to increasingly larger domains, but there are tasks in which observability is not the only concern. For instance, if other agents (including humans) influence the task in ways that cannot be characterized upfront, then the typical MDP formulation or even versions (e.g., interactive or decentralized) of POMDPs do not capture this issue [19]. In such scenarios, it may be more useful to synthesize the overall behavior as a composition of partially specified local behaviors - tuned to a localized interaction context - with choices between them in order to adapt to the changing environment and task [7]. The composition of local behaviors may depend on high level observations, and an approach that acknowledges the Non-Markovian nature of the problem is required.

In the literature, three methods have been used in Non-Markovian domains. The first one consists in applying direct RL to observations, relying on *eligibility traces* [13]. The second one is a local optimization algorithm, MCESP [11], made sound by a specific definition of the equation used for value prediction. The third one is pure search in policy space, dominated by policy gradient [2], a category of methods that search in the continuous space of stochastic policies. The first two methods make use of value functions, while the last one avoids them altogether. In this paper, we introduce a novel Reinforcement Learning algorithm to learn in Non-Markovian domains, that performs a global stochastic search in policy space. As such, it belongs to the third category, but it is also based on an action-value function, and uses it to store an estimated upper bound of each action's value to bias the search. Our control method makes decisions locally, and separately at each choice point, while the prediction method takes into account the long-term consequences of actions. In general N-MDPs (as well as in stochastic processes on top of POMDPs' observations) the reward, and therefore the decisions, may depend on the whole chain of observations and actions. In problems of practical interests, however, the choices can be efficiently made separately in many cases. The prediction method, nonetheless, must take the effect of subsequent choices into account.

We show experimentally how the space of deterministic policies can be searched effectively by exploiting action values' upper bounds locally and evaluating policies globally. We first introduce two simple domains, taken from the literature of partially observable processes, that allow for the comparison with other methods. Then, we apply our methodology to Keepaway [15], a challenging and more realistic domain for RL. Here, instead of relying on function approximation, we consider a partially specified behavior with coarse-grained choices. While

function approximation attempts to learn the best decision for each state, generalizing from *similar* ones (whose definition depends on the specific approximator), our method aims at learning, at specific choice points among high-level actions, which option performs best across all the situations that actually happen, biased by their frequency. Even if RL (in MDPs) provably converges to the optimal behavior, on Keepaway it does not do so in any reasonable time. We show how a behavior better than the ones reported in the literature can be obtained in a number of samples an order of magnitude smaller. The results suggest that, when the domain does not allow a compact Markovian description, giving up the Markov property in order to reduce the representation might, with the appropriate algorithms, provide a good behavior (possibly optimal in the given representation) much faster than methods that achieve optimality in the underlying MDP.

2 Background and Related Work

In the following, we first define the notation, and then describe the methods most closely related to our own.

2.1 Notation

A Markov Decision Process is a tuple $MDP = \langle S, A, T, \rho \rangle$ where:

- S is a set of *states*
- A is a set of *actions*
- $T : S \times A \times S \to [0, 1]$ is the transition function. $T(s, a, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a)$ is the probability that the current state changes from s to s' by executing action a. Since $T(s, a, \cdot)$ is a probability distribution, then $\sum_{s' \in S} T(s, a, s') = 1 \; \forall \; s \in S$ and $a \in A$. If $T(s, a, s') = \{0, 1\}$ the system is said to be *deterministic*, otherwise it is *stochastic*.
- $\rho : S \times A \times \mathbb{R} \to [0, 1]$ is the reward function. $\rho(s, a, r) = Pr(r_{t+1} = r | s_t = s, a_t = a)$ is the probability to get a reward r from being in state s and executing action a. Analogously to the transition function, $\rho(s, a, \cdot)$ is a probability density function and $\int_{\mathbb{R}} \rho(s, a, r) dr = 1$. If the reward function is defined over a discrete subset $P \subset \mathbb{N}$, ρ is a probability distribution and the reward is said to be *deterministic* if $\rho(s, a, r) = \{0, 1\} \; \forall s \in S$, $a \in A$, and $r \in P$.

The behavior of the agent is represented as a function $\pi : S \times A \to [0, 1]$ called a *stationary policy*, where $\pi(s, a)$ is the probability of selecting action a in state s. If $\pi(s, a) = \{0, 1\} \; \forall \, s \in S$ and $a \in A$ the policy is *deterministic*.

A policy π and an initial state s_0 determine a probability distribution $\mu(\omega)$ over the possible sequences $\omega = (\langle s_t, a_t, r_{t+1} \rangle, \, t \geq 0)$. Given such a sequence, we define the *cumulative discounted reward* as

$$R(\omega) = \sum_{t \geq 0} \gamma^t r_{t+1} \tag{1}$$

where $0 < \gamma \leq 1$ is the *discount factor*. If $\gamma = 1$ the reward is *undiscounted*, which is allowed only if the MDP is episodic (it has at least an absorbing state, which is never left once entered) otherwise the total reward could diverge.

Analogously to Markov Decision Processes, a Partially Observable MDP, is a a tuple $\langle S, A, T, \rho, Z, O\rangle$, where $\langle S, A, T, \rho\rangle$ is an underlying MDP whose current state is not directly accessible. Instead of perceiving an element from S, the agent is given an element of O, the set of *observations*, which relates to the underlying state through the function $Z : O \times A \times S \rightarrow [0, 1]$ such that $Z(o, a, s) = Pr(o|s, a)$ is the probability of observing o when executing a in s. We consider the case of POMDPs in which S, T, and Z are unknown. Ignoring the actual state space and considering the controllable stochastic process $\langle O, A, T_o, \rho_o\rangle$ over observations, the unknown distribution

$$T_o(o, a, o') = \sum_{s \in S} Pr(s|o) * \sum_{s' \in S} T(s, a, s')Z(o', a, s)$$

depends not only on the current observation, but also on the underlying actual state and on the system's dynamics. The actual state in turn depends on the initial state and on the policy followed. Analogously, the reward obtained after each observation does not only statistically depend on the observation alone, and is therefore Non-Markovian.

Given an N-MDP $= \langle O, A, T_o, \rho_o\rangle$, we aim at computing the deterministic stationary reactive policy $\pi(o) = a$ that maximizes the expected value of the cumulative discounted reward:

$$R = \sum_{s_0} \sum_{\omega} \mu(\omega|\pi, s_0)Pr(s_0)R(\omega)$$

In the following we summarize what has been proved for direct RL methods in these circumstances.

2.2 Previous Results for N-MDPs

In Markov Decision Processes sub-optimal policies are never fix-points of policy iteration, so that each step produces not only a different policy, but a better one. MDPs are, therefore, well suited to hill-climbing, since optimal policies, and only those, are equilibrium points in MDPs, while this is not true in general for N-MDPs. hPOMDPs constitute a class of POMDPs in which the history of observations and actions is enough to determine the current state. Pendrith and McGarity [10] analyze the stability of TD(λ) and first-visit Monte Carlo [16]. They prove the following results:

- if a first-visit MC is used for an hPOMDP where $\gamma = 1$, then the optimal observation-based policies will be learning equilibria.
- the previous result does not apply to $\gamma = [0, 1)$
- if a TD(λ) method of credit assignment is used for direct RL on a N-MDP, then for $\gamma < 1$ it is not guaranteed that there exists an optimal observation-based policy representing a learning equilibrium.

Perkins and Pendrith [12] carry this analysis further, and include the exploration policy explicitly. They prove that there exists a learning equilibrium for 1-step TD methods if the exploration policy is *continuous* in the action values, while most of the former analysis had been conducted with ϵ-greedy which is discontinuous. So, for instance, following SoftMax, that assigns to every action a probability according to a Boltzmann distribution:

$$Pr(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_{a' \in A} e^{Q(s,a')/\tau}} \tag{2}$$

both Sarsa and Q-learning have at least one action-value function that is a learning equilibrium. The parameter τ in Eq. 2 balances exploration and exploitation: the higher τ the more the agent is likely to select a sub-optimal action (according to the *current* value function). The results prove that there exists a fixed point with respect to the update rule and a continuous exploration policy, but do not prove that such a fixed point can actually be reached. Moreover, the presented results do not consider that the exploration may change, for instance letting τ tend to zero.

2.3 A Sound Local Algorithm

Perkins [11] redefined the value function to overcome the above difficulties with discounted problems.

Let $\mu_\pi(\omega)$ be the probability distribution over the possible trajectories determined by a policy π. The author splits the reward with respect to an observation o from one of these trajectories ω in:

$$\begin{aligned} V^\pi &= \mathbb{E}^\pi_{\omega \sim \mu} \left[R(\omega) \right] \\ &= \mathbb{E}^\pi_{\omega \sim \mu} \left[R_{pre-o}(\omega) \right] + \mathbb{E}^\pi_{\omega \sim \mu} \left[R_{post-o}(\omega) \right] \end{aligned} \tag{3}$$

where $R_{pre-o}(\omega)$ is the cumulative discounted reward before o is encountered in ω for the first time, while $R_{post-o}(\omega)$ is the reward after the first occurrence of o. In the following, we shall omit the subscript $\omega \sim \mu_\pi(\omega)$, but all traces are extracted from μ if not otherwise noted. The value of an observation-action pair $\langle o, a \rangle$, with respect to a policy π, is the value of the policy when π is followed everywhere except for o, in which a is executed instead. Such a policy is represented as $\pi \leftarrow \langle o, a \rangle$, and clearly $\pi = \pi \leftarrow \langle o, \pi(o) \rangle$. Its value is:

$$Q^\pi(o, a) = \mathbb{E}^{\pi \leftarrow \langle o,a \rangle} \left[R_{post-o}(\omega) \right] \tag{4}$$

This definition differs from the usual definition for MDPs in two important respects: (1) every time (and not just the first one) the observation o is encountered, the agent executes the action a; (2) the value of an observation-action pair is not the discounted return following o, but the expected discounted reward following o *at that point* of the trace.

While in MDPs the optimal policy is greedy with respect to the action-value function, this is not necessarily true for POMDPs. With the definition of the

value function just given, this property is retained to some extent. In particular, for all π and $\pi' = \pi \leftarrow \langle o, a \rangle$

$$V^{\pi} + \epsilon \geq V^{\pi'} \iff Q^{\pi}_{o,\pi(o)} + \epsilon \geq Q^{\pi}_{o,a}$$

Monte Carlo Exploring Starts for POMDPs (MCESP) is a family of algorithms, that make use of the value function of Equation 3 to compute a *locally optimal* policy. The gained capability to hill-climb brings about the theoretical guarantee of local optimality, at the cost of updating one state-action pair at a time. For this reason MCESP proved to be slower than $Sarsa(\lambda)$ in the domain presented in the original paper.

We define a new algorithm for stochastic search that retains some of the ideas behind MCESP, while attempting a biased global search. Our algorithm relies on the ability of Monte Carlo policy evaluation to estimate the current policy, and performs a form of branch and bound related to *confidence bounds* [1] for N-MDPs. Although stochastic policies could, in general, perform better on N-MDPs, we only search in the space of deterministic policies as we are interested in getting *good* policies in the shortest number of episodes possible. It has been shown how deterministic policies can often provide such behaviors in practice [8], and we provide more examples in the experimental section.

3 The Algorithm: SoSMC

The main idea behind the algorithm, Stochastic Search Monte Carlo (SoSMC), is based on the intuition that often a few bad choices disrupt the value of all the policies that include them. Taking those policies as if they were as valuable as any other just wastes samples. We would rather like to realize that those actions are not promising and not consider them unless we have tried all the other options. The strategy would consider all the policies with a probability proportional to how *promising* they are, which we believe is beneficial in at least two ways: (1) the algorithm reaches the optimal policy earlier; (2) during the phase of evaluation of those *promising* but suboptimal policies, the behavior is as good as the current information allows.

The algorithm (cf. Algorithm 1) is constituted by two parts: the *exploratory* phase and the *assessing* phase, as described in the next section.

3.1 Exploration: Gathering Information

The exploration initializes the Q-function to drive the execution in the subsequent phase. The aim of the initial exploration is to determine an upper bound for each state-action pair. For a number of episodes exp_length the agent chooses a policy according to some strategy Σ (e.g., uniformly at random), and in each pair $\langle o, a \rangle$ stores the highest value that any policy, going through $\langle o, a \rangle$, has ever obtained.

Consider the simple example of the N-MDP in Figure 1(a). This N-MDP has three states and four actions with a total of four policies. Let the reward returned

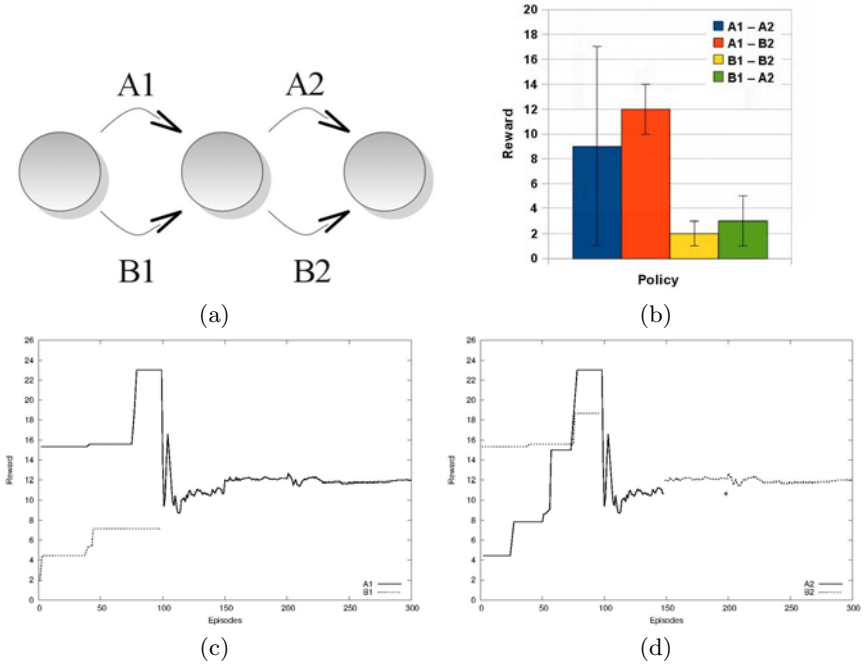


Fig. 1. A simple example of an N-MDP (a). The four policies return a reward normally distributed whose means and standard deviations are shown in (b). The evolution of the Q-function for the first state (actions A1 and B1) is represented in Figure (c), while for the second state (actions A2 and B2) is represented in Figure (d).

by each of those policies be normally distributed, with means and standard deviations represented in Figure 1(b). Figure 1(c) and 1(d) show the value of the Q-function for each action during a particular run. The first 100 episodes belong to the exploratory phase, in which the actions A1 and A2 obtain the highest reward, making the policy A1-A2 look particularly promising. An action is considered as *promising* as the highest value of the reward that choosing that action has ever given. In the case of A1-A2, its good result is due to the high variance, rather than the highest mean. This aspect will be addressed by the second phase of the algorithm.

The number of episodes in the exploratory phase should ideally allow for the sampling of each policy above its mean at least once. Depending on the particular strategy Σ and the shape of the distributions of the policies, such a number for *exp_length* might be computable. In practice, unless the problem is effectively hierarchically broken into much smaller problems, the number of episodes required is hardly feasible. In those cases, the exploration has to be shorter than what would be required to complete, and the algorithm will start with an upper bound for the limited number of policies visited, and keep exploring during the second phase.

Algorithm 1. SoSMC

$exp\_length \leftarrow$ number of episodes in the exploratory phase
$n \leftarrow$ current episode
$t \leftarrow$ last exploratory episode
$\alpha(n, o, a) \leftarrow$ learning step parameter
initialize $Q(s, a)$ pessimistically
{Exploratory phase}
for $i = 1$ to exp_length **do**
  generate a trajectory ω according to a policy π extracted from a strategy Σ
  for all $o \in O$, $a \in A$ $s.t.\langle o, a\rangle$ is in ω **do**
    $Q(o, a) = max(Q(o, a), R_{post-o}(\omega))$
  end for
end for
{Assessing phase}
for all other episodes : n **do**
  if n is such that the current estimate is considered accurate **then**
    $t = n$
    $\pi \leftarrow$ a policy chosen from Σ
  else
    $\pi \leftarrow$ the last policy chosen
  end if
  {Possible policy change after an exploratory episode}
  if $n = t + 1$ **then**
    $\pi \leftarrow$ the policy that greedily maximizes Q
  end if
  generate a trajectory ω from π
  if $n = t$ **then**
    for all $o \in O$, $a \in A$ $s.t.\langle o, a\rangle$ is in ω **do**
      $Q(o, a) = max(Q(o, a), R_{post-o}(\omega))$
    end for
  else
    for all $o \in O$, $a \in A$ $s.t.\langle o, a\rangle$ is in ω **do**
      $Q(o, a) = (1 - \alpha(n, o, a))Q(o, a) + \alpha(n, o, a)R_{post-o}(\omega)$
    end for
  end if
end for

If the domain does not allow for the estimation of a helpful upper bound, for instance because every action can potentially give high rewards, the first phase can be skipped initializing all actions optimistically. We conjecture that this may happen on synthetic domains in which the stochasticity is artificially injected, but it is rarer in real-world applications.

3.2 Assessment

We want to maximize the *expected* cumulative discounted reward, rather than the maximum obtainable one, therefore an evaluation of the *promising* policies is needed.

We rely on the main result behind stochastic search, reported (for minimization, but valid for maximization problems as well) in the following theorem [14]:

Theorem 1. *Suppose that θ^* is the unique minimizer of the loss function L on the domain Θ where $L(\theta^*) > -\infty$, $\theta_{new}(k)$ is the sample probability at iteration k, $\hat{\theta_k}$ is the best point found up to iteration k, and*

$$\inf_{\theta \in \Theta, \|\theta - \theta^*\| \geq \eta} L(\theta) > L(\theta^*)$$

for all $\eta > 0$. Suppose further that for any $\eta > 0$ and for all k there exists a function $\delta(\eta) > 0$ such that

$$Pr(\theta_{new}(k) : L(\theta_{new}(k)) < L(\theta^*) + \eta) > \delta(\eta)$$

Then, executing a random search with noise-free loss measurements, $\hat{\theta_k} \to \theta^*$ almost surely as $k \to \infty$.

In order to guarantee that the conditions expressed by the theorem are met we: (1) limit the search space to the *finite* set of deterministic stationary policies; (2) require immediate rewards to be bound; (3) require that the strategy Σ according to which the policies are picked assigns a non-zero probability to each of them. In particular, the second condition holds as the search space is finite, and the probability to land arbitrarily close to the optimal policy is non-zero, as such is the probability to land directly on it. If the optimal solutions are more than one, the algorithm might not converge on any single of them, but rather oscillate among optimal solutions. Since the value of optimal solutions is, in general, not known in advance, there is no obvious stopping criterion. In practice, and as it is common in stochastic search, we may define a threshold above which we accept any solution.

In the second phase the algorithm picks a policy according to Σ, evaluates it with first-visit Monte Carlo, and stops if the policy's reward is above a threshold. Monte Carlo methods wait until the end of the episode to update the action-value function, therefore the task needs to be episodic. The novel aspect of SoSMC is the way in which the search is biased. Reinforcement learning algorithms can traditionally be considered as composed by prediction and control. While the prediction part is borrowed from the literature (first-visit MC) the control part is based on the estimate of upper bounds, their storage in the action-value function, and their use to generate the next policy to try. Moreover, differently from MCESP, it performs a global search. It also employs a *consistent* exploration [3], that is, during the same episode, every time the agent perceives an observation it performs the same action.

If the reward is deterministic a single evaluation per policy is sufficient. Such a case may, for instance, occur on POMDPs in which the underlying MDP is deterministic and there is a single initial state. If the reward is stochastic, on the other hand, the capability to have an accurate estimate of the reward depends on the distribution. For some distributions it may be possible to compute confidence bounds and ensure that the reward returned by a policy is higher than

the threshold with some probability. In general, the estimated mean cannot be guaranteed to be correct after any finite number of samples. In such cases, we use a fixed number of samples k which empirically proves to be reliable. Although losing some of the theoretical guarantees, we experimentally show how SoSMC can outperform the best results in the literature for different domains. While an inaccurate estimation of a policy may deceive the stopping criterion, if the threshold is not too tight on the optimal value a *good* policy is in practice always found in the domains we have used.

The example of Figure 1 shows an assessment phase with $k = 50$. Beginning with episode 101, in each state the action with the highest value is locally selected. The policy A1-A2 is initially executed for k episodes and the values of the actions converge to their means. Every k episodes a new policy is generated at random, with its probability depending on action values, and executed to be evaluated. Notice how in the second phase only half of the policies are actually sampled, as the action B1 is never executed after the initial phase.

3.3 Exploration Strategies

Different choices are possible for the exploration strategy Σ, making SoSMC a family of algorithms. We have used both ϵ-greedy and SoftMax (cf. Equation 2). In the case of ϵ-greedy (where we refer to the algorithm as ϵ-SoSMC), the choice has been made for each state the first time it is encountered, and then remembered throughout the episode. Notice that this has not been applied to Sarsa, in which a separate decision is made if the same state is encountered multiple times in the same episode. The policies closest to the current optimal one are more likely to be selected, and become less and less probable as the distance from the optimal policy increases. As for SoftMax, again the current optimal policy is the most likely to be selected, but the neighbourhood is considered not just in the distance from such a policy, but also in the value of its actions, evaluated locally. SoSMC with SoftMax, referred to as Soft-SoSMC, performs particularly well in those domains in which the combinatorial aspect is minimal, and the choices can often be made separately.

4 Experimental Evaluation

In this section, we show the results of the application of our algorithm to three different domains. The first domain is the same one used for MCESP. In all of the domains, we have compared our algorithm with Sarsa as it is still the most effective method based on value functions on N-MDPs [9,11]. As they are not based on value functions and search for a stochastic policy, rather than a deterministic one, we also leave policy gradient methods out of the evaluation.

4.1 Parr and Russell's Grid World

This small grid world has been used as a test domain by several authors [9,11]. It has 11 states (Figure 2 (a)) in a 4 by 3 grid with one obstacle. The agent starts

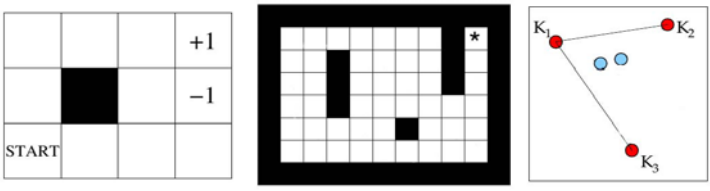


Fig. 2. The three domains used. (a) Parr and Russell's grid world, (b) Sutton's grid world, (c) Keepaway

at the bottom left corner. There is a target state and a penalty state whose rewards are +1 and -1 respectively. Both are absorbing states, that is when the agent enters them the episode terminates. For each action that does not enter a target state, the agent receives a reward of -0.04. The actions available in every state are `move north`, `move south`, `move east`, and `move west` which succeed with probability 0.8. With probability 0.1 the agent moves in one of the directions orthogonal to the desired one. In all of the previous cases if the movement is prevented by an obstacle the agent stays put. In any state the agent can only observe the squares east and west of it, having a total of four possible observations. In order to make the task episodic, the maximum number of actions allowed is 100, after which the environment is reset.

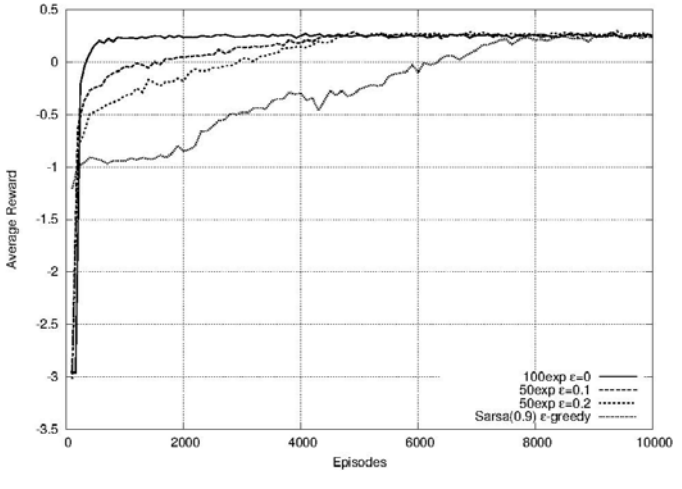


Fig. 3. Results for Parr and Russell's domain

Figure 3 shows the results of ϵ-SoSMC with an initial phase of 100 episodes and no exploration afterward (employing a constant $\alpha = 0.05$ and selecting the greedy policy at any time), with an initial phase of 50 episodes and exploration in the second phase ($\epsilon = 0.1$), and finally Sarsa(λ) with ϵ-greedy, in which ϵ starts

at 0.2 and decreases to zero in 80000 actions as described by Loch and Singh [9]. The threshold has been posed close to the optimum at 0.2. Each point is the average of the reward collected by the agent in the previous 100 episodes, and over 100 runs, including the exploration. It can be noted how SoSMC converges much faster than Sarsa and reaches the optimal solution reliably.

4.2 Sutton's Grid World

Sutton's grid world is a 9 by 6 grid with several obstacles and a goal state in the top right corner. It is accessible to the agent only through its 8 neighboring states, making it a POMDP. Only 30 possibly observations are actually possible in the grid world, and the initial state is chosen at every episode uniformly at random. The actions are the same as the previous domain, but they are deterministic. For this reason, only those that do not run directly into an obstacle are available. In order to make the task episodic we set the maximum number of actions in any given episode to 20. After each action the agent receives a reward of -1, except for when it enters the goal state, in which case it is 0. Every 200 episodes we pause the learning and take a sample, from each initial state, of the current best policy, whose average reward per episode is plotted in Figure 4.

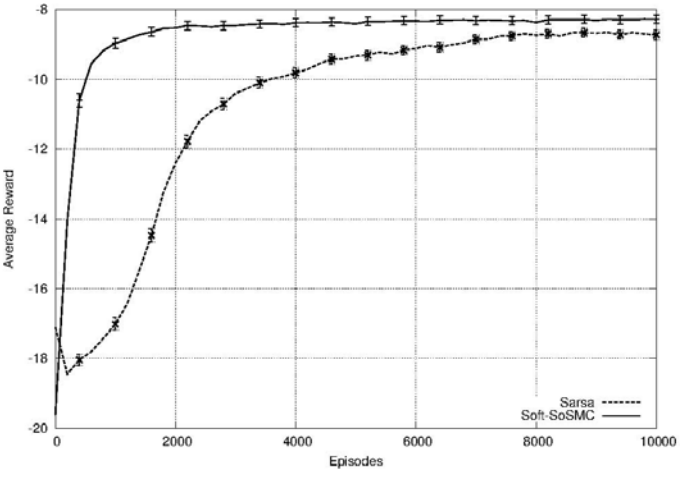


Fig. 4. Results for Sutton's domain

In this domain ϵ-SoSMC and Soft-SoSMC obtained similar results, therefore we only show Soft-SoSMC. We used SoftMax with no initial phase. The value function has been optimistically initialised and SoSMC launched from its second phase. In this experiment $\tau = 4$ and the algorithm explores every 20 episodes. The threshold is at -8.3. The results show how Soft-SoSMC finds, on average, a policy better than Sarsa. Sarsa obtains a value slightly, but statistically significantly smaller, as shown by the 95% confidence intervals which have been plotted on the graph every three points in order to not clutter the image.

4.3 Keepaway

Keepaway is a subtask of RoboCup Soccer in which one team, the *keepers*, must keep possession of the ball in a limited region as long as possible while another team, the *takers*, tries to gain possession. The task is episodic, and one episode ends whenever the takers manage to catch the ball or the ball leaves the region. We conducted our experiments on the 3 vs 2 task, i.e., with three keepers and two takers. Two different procedures are defined: `hold` and `pass(k)`. `Hold` keeps possession of the ball until the next decision can be made, while `pass(k)` passes the ball to the k-th team mate, where the team mates are sorted by their distance to the agent. A reward of 1/10 is collected every 1/10 of second, so that the agent maximizes the duration of the episode. We devise a representation with three variables: the two distances between the takers and the lines of pass towards the team mates, and the distance between the agent and the closest taker. Moreover, for each variable we consider only one threshold, having 8 observations in total. The simulator is provided with a policy, written by hand, meant as a benchmark. On our system, such a policy holds the ball for almost 16 seconds on average. We set the threshold for the stochastic search at 18 seconds. We also fix the behavior of two agents at a policy that waits for the takers to be withing a certain distance, and passes the ball to the team mate whose line of pass is farther from the opponents. The third agent is the one that is going to learn. Figure 5 shows the mean hold time per episode of Soft-SoSMC and Sarsa(λ). Each point is the average of the previous 200 episodes over 20 runs and *includes* the exploration (it is not just the best policy at any given time). Since this simulation is quite time consuming, we could not perform the same number of runs as the previous domains, in which we made sure that the confidence interval

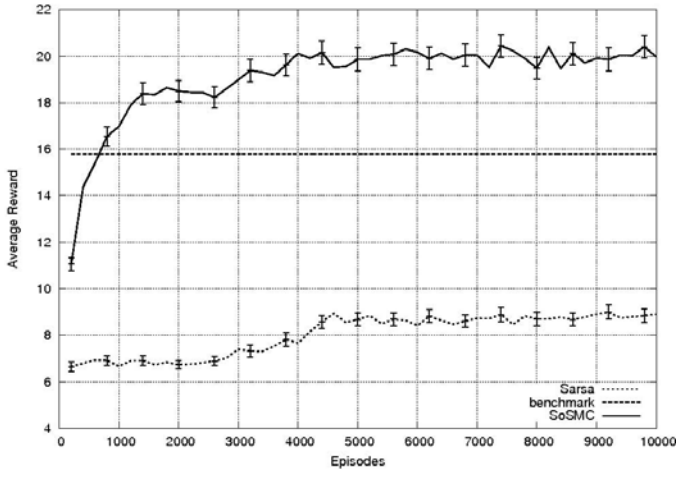


Fig. 5. Results for Keepaway. The average hold time per episode (y axis) is expressed in seconds.

for each point was negligible. This plot is therefore less smooth, and as with the previous domain, we show the 95% confidence interval explicitly. The initial phase of Soft-SoSMC has a length of 100 episodes. While Sarsa(λ) only learns a behavior that on average holds the ball for 9 seconds, despite the small number of states, our algorithm reaches 16 seconds in less than 1000 episodes and goes up to 20s in the next 4000 episodes. This behavior outperforms several other methods on this domain, including policy search algorithms [18,17,4,5,6]. This is not just due to the small size of the representation, as Sarsa(λ) is not able to improve the agent's behavior of more than about 3 seconds. Note that the representation is certainly non-Markovian, as we have run Q-learning choosing actions at random, and instead of off-policy learning the optimal action-value function - as it is proved it would on an MDP - the value of all actions collapsed and where almost the same. In this N-MDP then, a direct RL method does not succeed in learning, and a specific algorithm like SoSMC can instead leverage the representation nonetheless.

5 Conclusions

In this paper, we have analyzed the advantages of using Non-Markovian processes in order to reduce the representation space and to achieve fast learning. Our work leads us to conclude that, in the domains analyzed in this paper, the attempt to enrich the domain description to make it Markovian can introduce high complexity for the learning process, such that algorithms that are proved to converge to optimal solutions in the long term do not provide any good results when the number of available samples is limited. On the other hand, smaller representations can indeed be more effective, since good solutions can be found within the limit of the available samples, with an adequate learning method.

References

1. Auer, P., Jaksch, T., Ortner, R.: Near-optimal regret bounds for reinforcement learning. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 21, pp. 89–96 (2009)
2. Baxter, J., Bartlett, P.L.: Infinite-horizon policy-gradient estimation. Journal of Artificial Intelligence Research 15(4), 319–350 (2001)
3. Crook, P.A.: Learning in a state of confusion: Employing active perception and reinforcement learning in partially observable worlds. Technical report, University of Edinburgh (2006)
4. Jung, T., Polani, D.: Learning robocup-keepaway with kernels. In: JMLR: Workshop and Conference Proceedings (Gaussian Processes in Practice), vol. 1, pp. 33–57 (2007)
5. Kalyanakrishnan, S., Stone, P.: An empirical analysis of value function-based and policy search reinforcement learning. In: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2009, SC 2009. International Foundation for Autonomous Agents and Multiagent Systems, Richland, pp. 749–756 (2009)

6. Kalyanakrishnan, S., Stone, P.: Learning Complementary Multiagent Behaviors: A Case Study. In: Proceedings of the 13th RoboCup International Symposium, pp. 153–165 (2009)
7. Leonetti, M., Iocchi, L.: Improving the performance of complex agent plans through reinforcement learning. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, vol. 1, pp. 723–730 (2010)
8. Littman, M.L.: Memoryless policies: Theoretical limitations and practical results. In: From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior, pp. 238–247 (1994)
9. Loch, J., Singh, S.: Using eligibility traces to find the best memoryless policy in partially observable Markov decision processes. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 323–331 (1998)
10. Pendrith, M.D., McGarity, M.: An analysis of direct reinforcement learning in non-markovian domains. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML), pp. 421–429 (1998)
11. Perkins, T.J.: Reinforcement learning for POMDPs based on action values and stochastic optimization. In: Proceedings of the National Conference on Artificial Intelligence, pp. 199–204 (2002)
12. Perkins, T.J., Pendrith, M.D.: On the existence of fixed points for Q-learning and Sarsa in partially observable domains. In: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 490–497 (2002)
13. Singh, S.P., Sutton, R.S.: Reinforcement learning with replacing eligibility traces. In: Recent Advances in Reinforcement Learning, pp. 123–158 (1996)
14. Spall, J.C.: Introduction to Stochastic Search and Optimization, 1st edn. John Wiley & Sons, Inc., New York (2003)
15. Stone, P., Sutton, R.S., Kuhlmann, G.: Reinforcement learning for RoboCup-soccer keepaway. Adaptive Behavior 13(3), 165–188 (2005)
16. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
17. Taylor, M.E., Whiteson, S., Stone, P.: Transfer via inter-task mappings in policy search reinforcement learning. In: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2007, pp. 1–37. ACM, New York (2007)
18. Whiteson, S., Kohl, N., Miikkulainen, R., Stone, P.: Evolving soccer keepaway players through task decomposition. Machine Learning 59, 5–30 (2005), doi:10.1007/s10994-005-0460-9
19. Young, H.: Strategic learning and its limits. Oxford University Press, USA (2004)

Analyzing Word Frequencies in Large Text Corpora Using Inter-arrival Times and Bootstrapping

Jefrey Lijffijt, Panagiotis Papapetrou, Kai Puolamäki, and Heikki Mannila

Department of Information and Computer Science, Aalto University,
Helsinki Institute for Information Technology (HIIT), Finland
{firstname.lastname}@aalto.fi

Abstract. Comparing frequency counts over texts or corpora is an important task in many applications and scientific disciplines. Given a text corpus, we want to test a hypothesis, such as "word X is frequent", "word X has become more frequent over time", or "word X is more frequent in male than in female speech". For this purpose we need a null model of word frequencies. The commonly used bag-of-words model, which corresponds to a Bernoulli process with fixed parameter, does not account for any structure present in natural languages. Using this model for word frequencies results in large numbers of words being reported as unexpectedly frequent. We address how to take into account the inherent occurrence patterns of words in significance testing of word frequencies. Based on studies of words in two large corpora, we propose two methods for modeling word frequencies that both take into account the occurrence patterns of words and go beyond the bag-of-words assumption. The first method models word frequencies based on the spatial distribution of individual words in the language. The second method is based on bootstrapping and takes into account only word frequency at the text level. The proposed methods are compared to the current gold standard in a series of experiments on both corpora. We find that words obey different spatial patterns in the language, ranging from bursty to non-bursty/uniform, independent of their frequency, showing that the traditional approach leads to many false positives.

Keywords: burstiness, sequence analysis, natural language modeling.

1 Introduction

Analyzing word frequencies is important in many application domains, such as data mining and corpus linguistics. Suppose we have a set of texts and we want to test a hypothesis, such as "word X is frequent", "word X has become more frequent over time", or "word X is more frequent in male than in female speech". For such tasks, we need to have a null model of word frequencies. The standard for statistical testing of such hypothesis is based on the bag-of-words assumption, i.e., every word can occur at any position in a text with equal probability.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 341–357, 2011.

This assumption has been pervasively used by both data mining [15] and linguistics communities [5] for finding words with significantly elevated occurrences in a text. We show in this paper that for almost no word, its frequency distribution observed in text corpora corresponds to a binomial distribution. Thus, the binomial distribution is almost always an inappropriate null model for word frequency distribution.

In linguistics, frequencies of words and other phenomena such as proverbs, semantic tags, n-grams, etc., are widely used to study how people communicate. It is well known that the bag-of-words model is a poor descriptor of word occurrences. Linguists have gone as far as claiming that hypothesis testing of word frequencies is rarely useful to finding associations, and often leads to misleading results [14]. Others have noted that a measure of *dispersion* is necessary to improve significance testing [21], or that each significant result should be checked using an effect size measure [9] or manual investigation [20].

In information retrieval, the fraction of documents where a word occurs is used to detect content-related words. The *inverse document frequency*, used in the classic *tf-idf*, or more recent approaches such as *Okapi BM25* [2,22], is useful because content-related words are less dispersed than words with a grammatical function. Such models implicitly assume the bag-of-words setting. Usually the statistical significance of word frequencies is of no interest, because the task is not to find or study individual words that describe the documents but to rank documents according to their relevance to a given set of query words. Our problem setting, however, is very different and thus not directly comparable.

Our Approach. Comparing frequency counts over texts or corpora is an important task in many applications and scientific disciplines. The commonly used bag-of-words model, which can be described as a Bernoulli process with fixed rate, does not account for any structure present in natural languages. It can be easily shown that words have very different behavior in language, even at the word frequency level. In Figure 1, we illustrate the frequency histograms of the words *for* and *i* (lowercase *I*) in the British National Corpus [24]. These words are both very frequent, and approximately equally frequent. Yet, their frequency distribution is very different, thus employing the bag-of-words model in this example would be misleading.

Contextual behavior of words varies in language and is affected by several factors, such as genre, topic, author (gender, age, social class) etc. For example, in written language, especially in newspaper texts, there is avoidance of repeating a word, due to stylistic ideals, whereas in conversation, priming of words and syntactic structures plays an important role [10,23]. Hence, it is evident that natural language is non-homogeneous. There is great variance in word frequencies which depends on the specific word.

To model the natural behavior of words, we study their distribution throughout texts. The essential unit here is the interval between two occurrences of a word. We refer to this interval as the *inter-arrival time* between two instances. A recent study suggests that *inter-arrival times* in natural language can be modeled to a good accuracy using a Weibull distribution [1]. This parametric

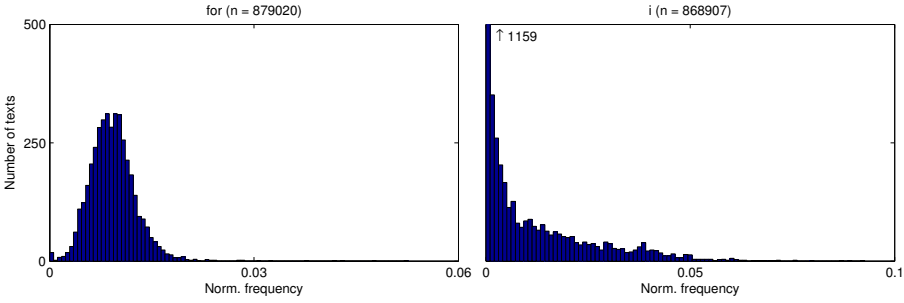


Fig. 1. Histogram of normalized frequencies vs. number of texts for the words *for* and i in the British National Corpus

distribution gives rise to a parameter β that can be interpreted as the *burstiness* of a word; we show this has a direct effect on the word frequency distribution. Bursty words tend to exhibit long inter-arrival times followed by short inter-arrival times, while the inter-arrival times for non-bursty words have smaller variance. The lower the burstiness parameter, the burstier the word: for example, $\beta_{for} = 0.93$ and $\beta_i = 0.57$.

Our Contributions. We propose two methods for modeling word frequencies that both take into account the behavior patterns of words. The first method is based on the inter-arrival time distribution of individual words. The second model is based on bootstrapping and takes into account only word frequency at the text level. We compare these methods to the current gold standard in a series of experiments on two large corpora: the British National Corpus (BNC) [24] and the San Francisco Call Newspaper Corpus (SFCNC) [17]. These corpora contain about 100 million and 63 million words, respectively. The experiments are based on comparing word frequencies over writing styles in the BNC and over time in SFCNC.

We show that taking the behavior of individual terms into account matters: in many cases it increases the frequency thresholds for the word to be reported as significantly frequent and therefore reduces the number of reported words. In addition, we find that the inter-arrival distribution can be used to give good predictions for the word-frequency distribution and that the inter-arrival and bootstrap methods give similar results.

2 Related Work

Word frequencies have been studied and analyzed in several domains. Research on graphs and networks has shown that many natural phenomena and patterns in human activity exhibit power-law behavior [3,7,16,18]. The discovery of power-law statistics occurred in the study of natural language; Zipf's law [26], relating the rank of words and their frequencies, describes the oldest known example of a power-law. It is surprising that for word frequencies in text documents, no such heavy-tailed modeling has been attempted.

The Bernoulli model has been widely used in modeling text in both data mining and linguistics. Dunning et al. [5] adopts the bag-of-words model to assess the statistical significance of word frequencies in text, assuming a Multinomial distribution, while Kleinberg [15] assumes multiple levels of frequency rates in text, where bursts of low frequencies may contain bursts of higher frequencies.

A significant amount of work has focused on detection of bursty structure in text, where bursty words are clustered to represent topics [8,12], or they are classified based on their frequency trajectories [11]. Additional work includes burstiness detection methods for query logs [25] or streams [13]. A burstiness-aware search framework has been introduced by Lappas et al. [17] which is fully non-parametric. All these methods, however, do not perform any significance assessment of word frequencies, thus they are orthogonal to the work presented in this paper.

Several effects of contextual behavior of words have been addressed in linguistics, such as text genre differences [4], word priming in conversation [10,23], differences in language use between males and females, age groups, and social classes [21]. Recent work by Altmann et al. [1] has shown that the distribution of successive occurrences of words can be modeled by the Weibull distribution, which is used in this paper.

3 Problem Setting

Let $\mathcal{S} = \{S_1, \ldots, S_{|\mathcal{S}|}\}$ be a corpus, i.e., a set of n texts, defined over a lexicon $\Sigma = \{q_1, \ldots, q_{|\Sigma|}\}$. Each text $S_i = w_1 \ldots w_{|S_i|}$ is a sequence of words, with $w_j \in \Sigma$ for $j = 1, \ldots, |S_i|$.

The frequency $\text{freq}(q, S_i)$ of a word q in a document S_i is the number of occurrences of q in S_i; the frequency of a word q in a corpus $\mathcal{S}$ is the total number of occurrences of q in $\mathcal{S}$, i.e., $\text{freq}(q, \mathcal{S}) = \sum_{i=1}^{n} \text{freq}(q, S_i)$. The size $\text{size}(\mathcal{S})$ of a corpus $\mathcal{S}$, is the total number of words in it, which is the sum of the lengths of all texts, i.e., $\text{size}(\mathcal{S}) = \sum_{i=1}^{n} |S_i|$.

We focus on assessing the statistical significance of word frequencies in texts. Given a word q and a corpus $\mathcal{S}$, we would like to decide whether the frequency of q is significantly higher in $\mathcal{S}$ than in some given corpus $\mathcal{T}$ that conveys background knowledge on the word frequency distribution. For this purpose, we define an appropriate model for probability and use the one-tailed p-value:

$$p(q, \mathcal{S}, \mathcal{T}) = Pr\left(\frac{\text{freq}(q, \mathcal{S})}{\text{size}(\mathcal{S})} \leq \frac{\text{freq}(q, \mathcal{T})}{\text{size}(\mathcal{T})}\right). \tag{1}$$

We are interested in words for which this p-value is less than a user-defined significance threshold $\alpha \in [0, 1]$:

Definition 1 *(Dominant word). Given a word q, a corpus $\mathcal{S}$, a background corpus $\mathcal{T}$, a p-value function p, and a significance threshold α, q is a* dominant word *in $\mathcal{S}$ if and only if*

$$p(q, \mathcal{S}, \mathcal{T}) \leq \alpha. \tag{2}$$

We consider the following two problems:

Problem 1. *Given a word q and two corpora $\mathcal{S}$, $\mathcal{T}$, decide whether q is a dominant word in $\mathcal{S}$, given $\mathcal{T}$.*

For example, $\mathcal{S}$ can include articles written by male authors and $\mathcal{T}$ articles written by female authors. Given a word q, we would like to assess the significance of the frequency of q in $\mathcal{S}$ compared to the frequency in $\mathcal{T}$. In other words, we would like to determine whether q is used by males at a significantly higher rate than by females.

Problem 2. *Given two corpora $\mathcal{S}$ and $\mathcal{T}$, find the set of words $\mathcal{Q} \subseteq \Sigma$, such that each $q_j \in \mathcal{Q}$ is a dominant word in $\mathcal{S}$, given $\mathcal{T}$.*

For example, $\mathcal{S}$ may include newspaper articles written, e.g., in one year, while $\mathcal{T}$ may include newspapers written over some previous years. Our task then is to detect all dominant words for that year (set $\mathcal{S}$) compared to the previous years (set $\mathcal{T}$). Using this set of words we may infer the most important topics during that year and also observe gradual change of the language.

Note that we allow the case where both $\mathcal{S}$ and $\mathcal{T}$ contain only a single text. In the experiments in Section 5 we show that even when $\mathcal{S}$ consist of only one text, taking into account the structure of the language is meaningful and our approach gives results that differ substantially from the bag-of-words model.

4 Methods

In Section 4.1 we briefly discuss the baseline method, whereas in Section 4.2 we introduce the method based on inter-arrival times. This method comes in two flavors: fully non-parametric or using the Weibull distribution to model inter-arrivals. In Section 4.3 we introduce the bootstrapping method.

4.1 Method 1: Bernoulli Trials

A popular method for significance testing in frequency comparison is based on the assumption that a word occurs at any position in a text with equal probability. This setting is modeled by a repetition of Bernoulli trials, and the frequencies then follow a binomial distribution. The binomial distribution can be accurately approximated with the Poisson distribution for faster computation. Computational methods for these distributions are available in any modern statistical software package. This method serves as the baseline for comparison.

Let $p_{q,\mathcal{S}}$ denote the probability of observing q at any position in $\mathcal{S}$. The probability of observing q exactly k times after n trials is given by the probability mass function of the binomial distribution:

$$f(k; n, p_{q,\mathcal{S}}) = \binom{n}{k} {p_{q,\mathcal{S}}}^k (1 - p_{q,\mathcal{S}})^{n-k}. \tag{3}$$

Let $\hat{p}_{q,\mathcal{T}}$ denote the empirical probability of observing q at any position in $\mathcal{T}$: $\hat{p}_{q,\mathcal{T}} = \text{freq}(q, \mathcal{T})/\,\text{size}(\mathcal{T})$. Since the null hypothesis is that $p_{q,\mathcal{S}} = p_{q,\mathcal{T}}$, we can

use $\hat{p}_{q,\mathcal{T}}$ as an estimate for $p_{q,\mathcal{S}}$. The p-value for the Bernoulli model is then given by setting $n = \text{size}(\mathcal{S})$, $p = \hat{p}_{q,\mathcal{T}}$ and summing over $k \in [freq(q,\mathcal{S}), n]$:

$$p_1(q,\mathcal{S},\mathcal{T}) = \sum_{k=\text{freq}(q,\mathcal{S})}^{\text{size}(\mathcal{S})} \binom{\text{size}(\mathcal{S})}{k} \hat{p}_{q,\mathcal{T}}^{k} (1-\hat{p}_{q,\mathcal{T}})^{\text{size}(\mathcal{S})-k}. \tag{4}$$

Function $p_1(q,\mathcal{S},\mathcal{T})$ gives the one-tailed p-value of observing a frequency at least as high as $\text{freq}(q,\mathcal{S})$, given the size of $\mathcal{S}$ and the estimate $\hat{p}_{q,\mathcal{T}}$. Its value can be computed using the cumulative distribution function of Equation (4). The computational complexity of this approach is $\mathcal{O}(\text{size}(\mathcal{S}) + \text{size}(\mathcal{T}))$. For the remainder of this paper, this method will be denoted as `Bin`.

4.2 Method 2: Inter-arrival Times

This approach takes into account the natural behavior of words as expressed by inter-arrival times between words. The method is again based on computing a one-tailed p-value for observing a certain frequency or higher in $\mathcal{S}$, similar to the Bernoulli method. However, we do not assume that a word can occur at any position with fixed and equal probability. Instead, the probability of a word occurrence depends on the distance to the previous occurrence. Two null models are considered: the first is non-parametric and is based directly on the observed inter-arrival times, whereas the second is based on the Weibull distribution. First, we define inter-arrival times.

Inter-arrival Times. Let $\mathcal{S}$ be an ordered set of texts, which we concatenate to produce one long text $S = w_1 \dots w_{\text{size}(\mathcal{S})}$. For each word $q_i \in \Sigma$ with $n = freq(q_i,\mathcal{S})$, let $q_i^1, \dots, q_i^n$ denote the *positions* where q_i occurs in S, i.e., $q_i^l = j$ if and only if w_j is the l^{th} occurrence of q_i in S. The j-th *inter-arrival time* of word q_i, denoted as $a_{i,j}$, is given by

$$a_{i,j} = q_i^{j+1} - q_i^j \text{ , for } j = 1, \dots, n-1. \tag{5}$$

We take the inter-arrival time before the first occurrence of the word and after the last occurrence by considering $\mathcal{S}$ to be a ring. For simplicity, we define:

$$a_{i,n} = q_i^1 + |S| - q_i^n. \tag{6}$$

This ensures that the probability of observing the word is computed properly. Note there are as many inter-arrival times as words.

Empirical p-value. To obtain a null model and associated p-values, we use Monte Carlo simulation to create randomized texts, and compare the frequencies in the randomized texts against the observed frequency $\text{freq}(q,\mathcal{S})$.

Consider N random texts $\mathcal{R}_1, \dots, \mathcal{R}_N$, which have a size equal to $\mathcal{S}$: $\text{size}(\mathcal{R}_i) = \text{size}(\mathcal{S})$ for $i = 1, \dots, N$. We produce the random texts using a probability distribution for inter-arrival times learned from the background corpus $\mathcal{T}$. That

is, we construct a sequence of occurrences of word q by repeatedly sampling randomly from the set of inter-arrival times of q. We approximate the one-tailed p-value using the empirical p-value [19]:

$$\hat{p}_2(q, \mathcal{S}, \mathcal{T}) = \frac{1 + \sum_{i=1}^{N} I(\text{freq}(q, \mathcal{S}) \leq \text{freq}(q, \mathcal{R}_i))}{1 + N}, \tag{7}$$

where $I(\cdot)$ is the indicator function. We do not have to normalize the frequencies, since $\mathcal{S}$ and $\mathcal{R}_i$ are by definition of the same size.

Empirical Inter-arrival Time Distribution. The main step of the algorithm is to sample an inter-arrival time from the inter-arrival distribution, which we denote as $f(x)$. In the non-parametric case, we sample an inter-arrival time uniformly at random from the observed inter-arrival times, i.e., each observed inter-arrival time has equal chance of being chosen. This can be implemented by keeping a vector of inter-arrival times.

The first occurrence is treated separately. We can be at any point in any possible inter-arrival time at the beginning of the text. However, it is more likely we are at some point in a long inter-arrival than in a short one. To be precise, this is proportional to the length of the inter-arrival and thus we should sample uniformly from $g(x) = C \cdot x \cdot f(x)$, where C is a normalization constant such that $\sum_x C \cdot x \cdot f(x) = 1$. This gives us the current inter-arrival time, and within this inter-arrival time, any position is equally likely. Fast sampling from this distribution can be implemented by associating a normalized probability with each unique element in $f(x)$.

Random corpora produced using this Monte Carlo sampling procedure can be used to compute estimates for the one-tailed p-value. For the remainder of this paper, this method will be denoted as $\texttt{IA}_E$.

Weibull Inter-arrival Time Distribution. Recent work suggests that inter-arrival times between words can be modeled well using the Weibull (or stretched exponential) distribution [1]. It is shown that for almost any word the Weibull model fits the data much better than a Poisson distribution, as measured by the explained variance (R^2). Nonetheless, this recent study is mostly based on one data source: discussions on Google groups [1]. As far as we know, this result has not been validated by any other study. The comparison of the Monte Carlo simulation between the Weibull distribution and the empirical inter-arrival distribution will be the first evaluation of this result.

The probability density function for the Weibull distribution is given by

$$f(x) = \frac{\beta}{\alpha} \left(\frac{x}{\alpha}\right)^{\beta - 1} e^{-(x/\alpha)^{\beta}}, \tag{8}$$

where $\alpha, \beta > 0$ are the *scale* and the *shape* parameters, respectively. If $\beta = 1$, the Weibull distribution equals the Poisson distribution, and if $\beta \to 0$ it approaches a power-law and becomes heavy-tailed. We fit the parameters using the maximum-likelihood estimation. Implementations for fitting and sampling

for Weibull distributions are available in software packages for statistical analysis, such as R and Matlab.

The Monte Carlo simulation using the Weibull distribution is implemented as follows: sampling from $f(x)$ can be accomplished using standard statistical software, while the distribution $g(x) = C \cdot x \cdot f(x)$,

$$g(x) = C \cdot x \cdot f(x) = \frac{k}{\alpha \Gamma(1 + \frac{1}{\beta})} \left(\frac{x}{\alpha}\right)^{\beta} e^{-(x/\alpha)^{\beta}} \tag{9}$$

can be sampled by using the cumulative distribution function for $g(x)$,

$$G(x) = \int_0^a dx \cdot g(x) = 1 - \frac{\Gamma\left(1 + \frac{1}{\beta}, \left(\frac{a}{\alpha}\right)^{\beta}\right)}{\Gamma\left(1 + \frac{1}{\beta}\right)}. \tag{10}$$

Now, if we substitute $y = \left(\frac{a}{\alpha}\right)^{\beta}$, then $G(x)$ becomes the Gamma distribution with shape $k = 1 + 1/\beta$ and scale $\theta = 1$. We can sample a random number y from the Gamma distribution and obtain an inter-arrival time by $r = \lceil \alpha \cdot y^{1/\beta} \rceil$. The rounding is necessary because the Weibull distribution is continuous and > 0, while inter-arrival times are discrete and ≥ 1.

Again, using Equation (7) and the random samples obtained from this Monte Carlo simulation, we can compute an estimate for the one-tailed p-value. The computational complexity of this method is $\mathcal{O}(N \operatorname{size}(\mathcal{S}) + \operatorname{size}(\mathcal{T}))$ and the memory requirement is $\mathcal{O}(\operatorname{size}(\mathcal{T}))$. Since computations are done word-per-word, the memory cost can be reduced by storing only one vector of inter-arrival times at a time. For the remainder of this paper, this method will be denoted as $\texttt{IA}_W$.

4.3 Method 3: Bootstrapping

Instead of using inter-arrival times we can use a non-parametric approach to model the word frequency distribution directly.

Let $\mathcal{S}$ contain only one text, i.e., $\mathcal{S} = \{S\}$, and let $\mathcal{T}$ be a corpus with many texts. A straightforward approach to compute a p-value for the observed word frequency in $\mathcal{S}$ is to count the fraction of texts in $\mathcal{T}$ where the normalized frequency is larger. However, if $\mathcal{S}$ contains multiple texts, we would like to take into account heterogeneity between texts in both $\mathcal{S}$ and $\mathcal{T}$. We use bootstrapping [6] to approximate the p-value, although for this purpose also analytical estimates might be used.

The procedure is as follows: we take N random sets of texts $\mathcal{R}_1, \ldots, \mathcal{R}_N$, from the background corpus $\mathcal{T}$, each set having the same number of texts as $\mathcal{S}$: $|\mathcal{R}_i| = |\mathcal{S}|$. This leads to the problem that $\operatorname{size}(\mathcal{R}_i)$ is not necessarily equal to $\operatorname{size}(\mathcal{S})$, thus we should use normalized frequencies. We use the pooled frequency, divided by the pooled text size. Alternatively, one could use averages of frequencies that are normalized per text. Now, the empirical p-value (similar to Equation (7)) is

$$\hat{p}_3(q, \mathcal{S}, \mathcal{T}) = \frac{1 + \sum_{i=1}^{N} I\left(\frac{\operatorname{freq}(q,\mathcal{S})}{\operatorname{size}(\mathcal{S})} \leq \frac{\operatorname{freq}(q,\mathcal{R}_i)}{\operatorname{size}(\mathcal{R}_i)}\right)}{1 + N}. \tag{11}$$

The computational complexity of this method is $\mathcal{O}(N|\mathcal{S}| + \text{size}(\mathcal{S}) + \text{size}(\mathcal{T}))$ and the memory requirement is $\mathcal{O}(|\mathcal{T}|)$. For the remainder of this paper, this method will be denoted as `Boot`.

5 Experiments

The performance of our methods has been benchmarked on two large corpora:

The *British National Corpus* (BNC) [24] is the largest linguistically annotated corpus that is available in full-text format. It contains almost 100 million words of British English, spread over 4,049 texts, which are classified in text genres, such as fiction, academic prose, newspaper articles, transcribed conversation and more. The corpus is a result of careful digitization and has been annotated with meta information such as author gender, age, etc. and has been part-of-speech-tagged automatically with manual validation. We preprocess the data by removing all capitalization.

The *San Francisco Call Newspaper Corpus* (SFCNC) contains tokenized and stemmed newspaper articles published in the San Francisco Call, a daily newspaper, between 1900 and 1909, with stopwords removed. The SFCNC has been constructed and used by Lappas et al. [17]. The corpus consists of three periods:

- Period I: 110,387 articles published from 01/01/1900 to 31/12/1901.
- Period II: 133,684 articles published from 01/01/1903 to 31/12/1904.
- Period III: 129,732 articles published from 01/01/1908 to 31/12/1909.

The experiments are based on comparing word frequencies over writing styles in the BNC and over time in SFCNC. In Section 5.1, we present a simple proof of concept benchmark to show that taking into account individual behavior of words matters. We discuss the differences between male/female authors and four text-genres in the BNC in Sections 5.2 and 5.3. Significant language changes over time in the SFCNC are illustrated in Section 5.4 and the proposed methods also managed to detect dates of significant events.

5.1 BNC: A Simple Benchmark

We performed a simple benchmark on the BNC to show that *burstiness* matters when assessing the statistical significance of word frequencies. For simplicity, we used in this experiment a fixed text length of 2,000 words both for $\mathcal{S}$ and $\mathcal{T}$, which leaves us with 3,676 texts. We compared the significance thresholds for the most frequent words in the BNC and words with frequency just below 100,000. In detail, for each of the 30 words, we computed the word frequency that is required to make that word significant at the level $\alpha \leq 0.01$. Because the texts in the BNC are not ordered, we order them randomly.

In Table 1 we show the results of the proposed methods, $\texttt{IA}_\texttt{E}$, $\texttt{IA}_\texttt{W}$, and `Boot`, compared to `Bin`. Also, $\beta_{q,T}$ indicates the value of the shape parameter β of the Weibull distribution for each word q in $\mathcal{T}$. $\beta = 1$ corresponds to an exponential distribution, which we consider to be *non-bursty*. The lower the β the *burstier* the word. If $\beta > 1$, then the distribution is more regular than exponential, which we shall also consider to be *non-bursty*.

Table 1. Actual frequencies, parameters $\beta_{q,\mathcal{T}}$, and significance thresholds for the most frequent words in the BNC and words with frequency just below 100,000. Thresholds are computed for a text of length 2,000 words and $\alpha \leq 0.01$. $freq(q, \mathcal{T}$ is the frequency of the Word in the BNC. $\beta_{q,\mathcal{T}}$ is the burstiness of the word, given by the Weibull distribution. `Bin`, is the binomial model. $\texttt{IA}_E$ and $\texttt{IA}_W$ are the inter-arrival methods using empirical and Weibull distribution. `Boot` is the bootstrapping method. Largest differences occur when $\beta_{q,\mathcal{T}}$ is lowest.

Word	freq$(q,\mathcal{T})$	$\beta_{q,\mathcal{T}}$	Bin	$\texttt{IA}_E$	$\texttt{IA}_W$	Boot	Word	freq$(q,\mathcal{T})$	$\beta_{q,\mathcal{T}}$	Bin	$\texttt{IA}_E$	$\texttt{IA}_W$	Boot
the	6043900	1.10	149	152	143	197	how	99022	0.65	7	9	9	10
of	3043549	1.02	82	85	80	116	most	98051	0.77	7	8	8	7
and	2617864	1.08	72	72	70	95	back	96978	0.66	7	9	9	11
to	2594656	1.05	71	72	70	82	get	96000	0.60	7	10	10	20
a	2165365	1.01	61	63	61	72	way	95763	0.78	7	8	8	7
in	1938440	1.01	56	57	55	73	our	93272	0.53	7	11	10	17
that	1119422	0.87	35	40	38	69	down	92084	0.67	7	9	9	10
it	1054755	0.79	34	39	37	79	made	91383	0.80	7	8	7	8
is	990747	0.77	32	40	37	54	right	90533	0.57	7	10	9	38
was	881620	0.72	29	39	35	53	between	90519	0.70	7	8	8	8
for	879020	0.93	29	31	30	37	got	90165	0.51	7	12	12	20
i	868907	0.57	29	57	48	110	er	89845	0.43	7	28	26	54
's	784709	0.75	27	33	31	70	much	89842	0.79	7	7	8	7
on	729923	0.91	25	27	27	37	work	89344	0.61	7	9	9	11
you	667623	0.56	24	49	42	100	think	88665	0.56	7	11	10	17

The first observation we make concerns the six most frequent words (*the* – *in*), which have $\beta \geq 1.00$ and are thus non-bursty. The inter-arrival methods give similar frequency thresholds as the binomial model, although the bootstrapping method suggests that even for these words we should be more conservative in estimating p-values.

On the left side of the table are the words *for* and *i*, used previously in the example of Figure 1. The binomial model does not distinguish between the two words, while the three proposed methods do, by requiring a much higher frequency for *i* to be considered significant. The words on the right side of the table suggest there is difference between various words, regardless of frequency. Words such as *right* and *er*, but also *get*, *got*, and *think* are bursty and all three methods suggest we should assess the significance much more conservatively. Regarding the rest of the words in the table, we can conclude that both inter-arrival based methods perform similarly, with $\texttt{IA}_W$ consistently requiring slightly lower frequencies than $\texttt{IA}_E$. `Boot` often gives the highest threshold, but for few words (*most, way, much*) the results are similar to the binomial model.

5.2 BNC: Differences between Male and Female Authors

Next, we studied text variation between male and female authors in the BNC. For this experiment, we selected all fiction texts from BNC and split them into two groups: those written by males BNC_{male} and those written by females

Table 2. Number of dominant words for written fiction by male or female authors at various significance thresholds α. `Bin`, is the binomial model. $\texttt{IA}_E$ and $\texttt{IA}_W$ are the inter-arrival methods using empirical and Weibull distribution. `Boot` is the bootstrapping method. *Any* is the number of words reported as dominant by any of the methods. The inter-arrival and bootstrap methods show many of the words reported as significantly frequent by the binomial method are not significant. The inter-arrival method using Weibull distribution is most conservative.

Gender	α	`Bin`	$\texttt{IA}_E$	$\texttt{IA}_W$	`Boot`	*Any*	Gender	α	`Bin`	$\texttt{IA}_E$	$\texttt{IA}_W$	`Boot`	*Any*
Male	0.1%	183	133	110	119	185	Female	0.1%	202	147	123	131	202
Male	1.0%	264	210	182	186	266	Female	1.0%	290	222	195	210	290
Male	10.0%	417	375	359	366	417	Female	10.0%	470	420	400	405	471

BNC_{female}. We conducted two experiments: in the first we searched for dominant words in BNC_{male}, thus we set $\mathcal{S} = BNC_{male}$ and $\mathcal{T} = BNC_{female}$, and secondly we performed the reverse experiment. The performance of the proposed methods was compared to that of `Bin` for different significance thresholds α.

In Table 2, we can see the number of dominant words produced by each method for $\alpha = 0.1\%, 1.0\%, 10\%$. We also recorded the number of words detected as dominant by at least one of the methods, which is denoted as *Any* in the table. We can conclude that the number of dominant words detected by the three proposed methods are always less than those detected by `Bin` for both genders. For example, a significance threshold of 0.1%, the number of dominant words detected by `Bin` are approximately 1.7 times as many as those detected by $\texttt{IA}_W$, 1.4 times as many as those detected by $\texttt{IA}_E$ and 1.5 times as many as those detected by `Boot`. Also, $\texttt{IA}_W$ consistently detects the smallest number of dominant words. We also observed that dominant words detected by the proposed methods were nearly always flagged as dominant by `Bin`. Further investigation showed these words were reported by one of the inter-arrival methods and have p-values just above α for all other methods.

An overview of all p-values resulting from this experiment is given in Figure 2. The six displays compare all methods pairwise to each other. The in-sets enlarge the view at small p-values. We found that the inter-arrival time methods and `Boot` report *smoothed* p-values in many cases, i.e., p-values below 0.5 are higher and p-values above 0.5 become lower, in comparison to the binomial model. We find also that there is much agreement between $\texttt{IA}_E$ and `Boot`. The Weibull distribution appeared to give more variable results and larger differences compared to the binomial model than the other two methods. In general, the inter-arrival time method and `Boot` have greater agreement with each other than with the binomial model, as is clearly shown in the in-sets in all six figures.

5.3 BNC: Differences between the Main Genres

We studied text variation between the four main genres in BNC, i.e., *conversation*, *fiction prose*, *newspaper articles*, *academic prose*. Texts were split into four groups, one group per genre. Then for each genre, we set $\mathcal{S}$ to contain all

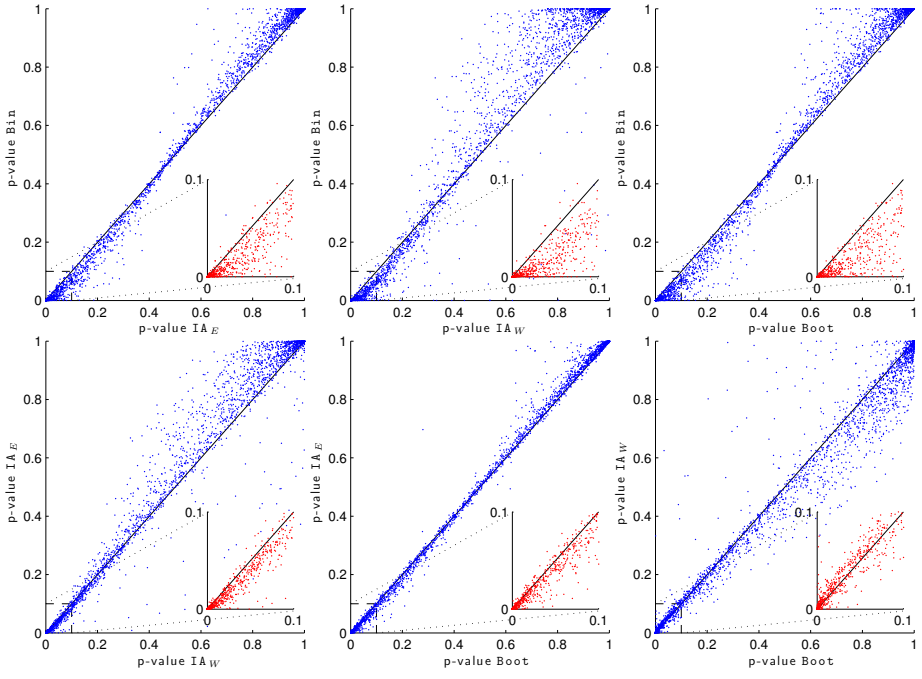


Fig. 2. Comparison of p-values between the four methods for male and female authors in the BNC. Each figure gives p-values from one method, against p-values in another method. Each point corresponds to a word. For explanation of labels see Table 2. The p-values from both experiments (male vs. female and vice versa) have been aggregated. The in-sets show more detail for the lower p-values < 0.1. We found that the binomial model gives very different results from all three other methods (top figures). The inter-arrival methods using empirical distribution and the bootstrap method show great agreement (bottom-centre figure). The inter-arrival method using Weibull distribution shows greater variance (bottom-right, bottom-left, and top-centre figure).

texts of that genre and $\mathcal{T}$ to contain the rest of the corpus. The performance of the proposed methods was compared to that of the binomial model for different significance thresholds α.

In Table 3, we can see the number of dominant words produced by each method and for each genre, for $\alpha = 0.1\%, 1.0\%, 10\%$. The behavior is the same as that observed for the male vs. female experiment. Again, we observed that nearly all dominant words detected by the proposed methods were also flagged as dominant by `Bin`. A figure illustrating the comparison of p-values is omitted due to space limitations. The results support the observations made in Figure 2.

5.4 SFCNC: Language Change over Time

We studied language variation between the three periods in SFCNC. For each period, we set $\mathcal{S}$ to contain all texts of that period and $\mathcal{T}$ to contain all texts from

Table 3. Number of words marked as dominant for each genre at various significance thresholds α. For explanation of labels see Table 2. The inter-arrival and bootstrap methods show many of the words reported as significantly frequent by the binomial method are not significant. The inter-arrival method using Weibull distribution and bootstrapping are most conservative. For *conversation* the differences between binomial and the other methods are smallest and for *news* they are greatest.

Genre	α	Bin	IA_E	IA_W	Boot	*Any*	Genre	α	Bin	IA_E	IA_W	Boot	*Any*
Conv	0.1%	381	328	308	314	381	News	0.1%	532	363	315	316	532
Conv	1.0%	412	384	363	367	412	News	1.0%	634	488	420	434	634
Conv	10.0%	473	453	447	446	474	News	10.0%	796	717	670	668	796
Fict	0.1%	505	388	339	352	507	Acad	0.1%	680	600	552	562	681
Fict	1.0%	573	496	446	464	573	Acad	1.0%	746	677	644	653	746
Fict	10.0%	682	629	619	610	682	Acad	10.0%	842	811	787	787	844

Table 4. Number of words marked as dominant for each news period at various significance thresholds α. For explanation of labels see Table 2. The inter-arrival and bootstrap methods show many of the words reported as significantly frequent by the binomial method are not significant. The inter-arrival method using Weibull distribution is most conservative. The differences between IA_E and Boot are small.

Period	α	Bin	IA_E	IA_W	Boot	*Any*	Period	α	Bin	IA_E	IA_W	Boot	*Any*
I	0.1%	141	73	50	73	141	II	10.0%	334	269	268	279	337
I	1.0%	229	144	113	134	231	III	0.1%	119	65	46	66	119
I	10.0%	423	339	346	340	428	III	1.0%	172	112	96	117	173
II	0.1%	113	41	19	46	113	III	10.0%	305	250	254	266	305
II	1.0%	182	99	74	98	182							

the other two periods. The performance of the proposed methods was compared to that of the binomial model for different significance thresholds α.

In Table 4, we can see the number of dominant words produced by each method and for each period, for $\alpha = 0.1\%, 1.0\%, 10\%$. The results at large are the same is in the experiment on the BNC. The differences between the proposed methods and the binomial model are even larger than before, especially at $\alpha = 0.1\%$. About half of the words marked as dominant by the binomial model, are false-positives according to the inter-arrival method using empirical distribution or bootstrap method. Using the Weibull distribution suggests even fewer truly significant words.

A full comparison of the p-values computed by all methods, aggregated over the three news periods, is shown in Figure 3. The in-sets show more detail for the lower p-values. Again, as in the BNC, the three proposed methods give higher p-values than Bin when $\alpha \leq 0.1$. In addition, for the same significance level, IA_W is clearly more conservative than the other methods. Also, the agreement between IA_E and Boot has decreased slightly, where IA_E gives slightly more conservative estimates.

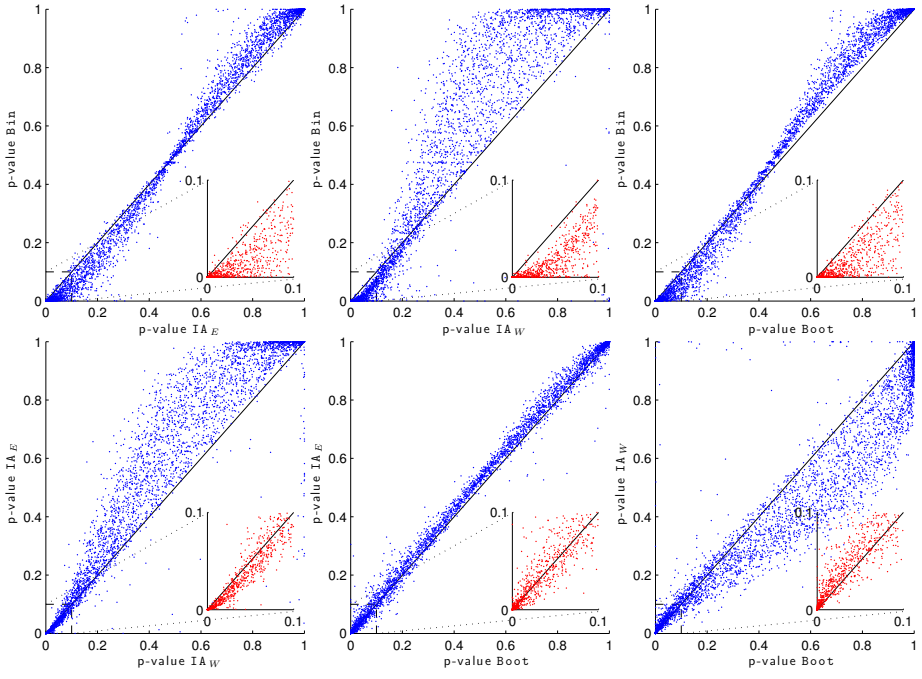


Fig. 3. Comparison of p-values between the four methods for the three periods in the SFCNC. Each figure gives p-values from one method, against p-values in another method. Each point corresponds to a word. For explanation of labels see Table 2. The p-values from all experiments have been aggregated. The in-sets show more detail for the lower p-values < 0.1. The figures confirm the findings of Figure 2. All three methods give more conservative p-values than binomial, and the pairwise differences between the inter-arrival time methods and the bootstrap method are similar to the genre experiment.

5.5 SFCNC: Locating Dates of Important Events

As a final test, we studied the intervals of word bursts reported in Lappas et al. [17]. These intervals correspond to bursts of some word after or around a significant historical event. We computed for each of the query words the days where this word is dominant, using $\alpha = 1\%$, and compare these to the corresponding intervals given by the search framework presented in their paper.

In Table 5 we find the results for one such query: *Jacksonville*. This interval (27 Apr–20 May) corresponds to the great fire at Jacksonville, Florida that occurred at May 3rd, 1901. We find that using any of the methods discussed in this paper find a shorter interval (5 May–8 May), and significant discussion one week later. The inter-arrival and bootstrap methods restrict the set of days even further. Due to lack of space the other results are omitted. In most cases the results were similar to the finding above, and for certain words, the intervals corresponded to those found in Lappas et al.

Table 5. Dates where the word *Jacksonville* occurs significantly frequent. `Lappas` is the method used in Lappas et al. [17]. `Bin`, is the binomial model. $\texttt{IA}_E$ and $\texttt{IA}_W$ are the inter-arrival methods using empirical and Weibull distribution. `Boot` is the bootstrapping method. An "x" corresponds to the word being dominant in the SFCNC at that day. All methods suggests stricter intervals than `Lappas` and the inter-arrival and boostrap methods flag the smallest sets of days.

	0427	0428	0429	0430	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511	0512	0513	0514	0515	0516	0517	0518	0519	0520
1901																								
`Boot`	.	.	.	.	.	.	.	.	x	x	.	x	.	.	.	.	.	x	.	.	x	.	.	.
$\texttt{IA}_W$	.	.	.	.	.	.	.	.	x	x	x	x	.	.	.	.	.	.	.	.	.	.	.	.
$\texttt{IA}_E$	.	.	.	.	.	.	.	.	x	x	.	x	.	.	.	.	.	x	.	.	.	.	.	.
`Bin`	.	.	.	.	.	.	.	.	x	x	x	x	.	.	.	.	x	x	.	.	x	.	.	.
`Lappas`	x	.	x	.	.	.	.	x	x	x	x	x	x	x	x	x	x	x	.	.	x	x	x	x

6 Conclusion

Models based on the bag-of-words assumption have been prevalent in text analysis and have been proven to perform well in a wide variety of contexts. The bag-of-words assumption provides a good estimate of the expected number of word occurrences in text. However, the variance—or more generally, the shape of the word frequency distribution—is seriously misestimated. We have introduced a method for assessing the significance of word frequencies that is based on the inter-arrival times. The method can use either the empirical distribution or a parametric distribution such as Weibull. By comparing the sets of dominant words given by the binomial model, the inter-arrival based method and the bootstrap-based method, we have shown that any statistical significance test on word occurrences that is based on the bag-of-words assumption tends to overestimate the significance of the observed word frequencies and hence result to false positives. Thus, bag-of-words based methods should not be used to asses the significance of word frequencies. One should either use an empirical method such as the bootstrap model presented in the paper, or the inter-arrival time based method.

An interesting direction for future work is to use the idea of inter-arrival times instead of bag-of-words in other scenarios, such as information retrieval, and to study test statistics other than word frequencies, which could be based on inter-arrival times directly. Also, further research on parametric distributions for inter-arrival times of words is warranted by the significant differences in the experimental results between the empirical and Weibull distribution.

Acknowledgements. This work was supported by the Finnish Centre of Excellence for Algorithmic Data Analysis Research (ALGODAN) and the Academy of Finland (Project 1129300). We thank Terttu Nevalainen, Tanja Säily, and Turo Vartiainen for their helpful comments and discussions.

References

1. Altmann, E.G., Pierrehumbert, J.B., Motter, A.E.: Beyond word frequency: Bursts, lulls, and scaling in the temporal distributions of words. PLoS ONE 4(11), e7678 (2009)
2. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman, Amsterdam (1999)
3. Barabási, A.-L.: The origin of bursts and heavy tails in human dynamics. Nature 435, 207–211 (2005)
4. Biber, D.: Dimensions of register variation: A cross-linguistic comparison. Cambridge University Press, Cambridge (1995)
5. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. Computational Linguistics 19, 61–74 (1993)
6. Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. Chapman and Hall/CRC (1994)
7. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: ACM SIGCOMM, pp. 251–262 (1999)
8. Fung, G.P.C., Pui, G., Fung, C., Yu, J.X., Yu, P.S., Yu, S., Lu, H.: Parameter free bursty events detection in text streams. In: VLDB, pp. 181–192 (2005)
9. Gries, S.T.: Null-hypothesis significance testing of word frequencies: a follow-up on Kilgarriff. Corpus Linguistics and Linguistic Theory 12, 277–294 (2005)
10. Gries, S.T.: Syntactic priming: A corpus-based approach. Journal of Psycholinguistic Research 34(4), 365–399 (2005)
11. He, Q., Chang, K., Lim, E.-P.: Analyzing feature trajectories for event detection. In: ACM SIGIR, pp. 207–214 (2007)
12. He, Q., Chang, K., Lim, E.-P.: Using burstiness to improve clustering of topics in news streams. In: IEEE ICDM, pp. 493–498 (2007)
13. He, Q., Chang, K., Lim, E.-P., Zhang, J.: Bursty Feature Representation for Clustering Text Streams. In: SIAM SDM, pp. 491–496 (2007)
14. Kilgarriff, A.: Language is never, ever, ever, random. Corpus Linguistics and Linguistic Theory 1-2, 263–275 (2005)
15. Kleinberg, J.: Bursty and hierarchical structure in streams. DMKD 7, 373–397 (2003)
16. Kumar, R., Novak, J., Raghavan, P., Tomkins, A.: On the bursty evolution of blogspace. World Wide Web 8(2), 159–178 (2005)
17. Lappas, T., Arai, B., Platakis, M., Kotsakos, D., Gunopulos, D.: On burstiness-aware search for document sequences. In: ACM SIGKDD, pp. 477–486 (2009)
18. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: Densification and shrinking diameters. ACM TKDD 1(1) (2007)
19. North, B.V., Curtis, D., Sham, P.C.: A note on the calculation of empirical p-values from Monte Carlo procedures. The American Journal of Human Genetics 71(2), 439–441 (2002)
20. Rayson, P., Garside, R.: Comparing corpora using frequency profiling. In: 38th ACL Workshop on Comparing Corpora, pp. 1–6 (2000)
21. Rayson, P., Leech, G., Hodges, M.: Social differentiation in the use of English vocabulary: some analyses of the conversational component of the British National Corpus. International Journal of Corpus Linguistics 2(1), 133–152 (1997)
22. Robertson, S.E., Walker, S.: Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In: ACM SIGIR, pp. 232–241 (1994)

23. Szmrecsanyi, B.: Language users as creatures of habit: A corpus-based analysis of persistence in spoken English. Corpus Linguistics and Linguistic Theory 1(1), 113–149 (2005)
24. The British National Corpus, version 3, BNC XML edn. (2007)
25. Vlachos, M.: Identifying similarities, periodicities and bursts for online search queries. In: ACM SIGMOD, pp. 131–142 (2004)
26. Zipf, G.K.: Human behavior and the principle of least effort. Addison-Wesley, Reading (1949)

Discovering Temporal Bisociations for Linking Concepts over Time

Corrado Loglisci and Michelangelo Ceci

Dipartimento di Informatica, Università degli Studi di Bari "Aldo Moro"
via Orabona, 4 - 70125 Bari - Italy
{loglisci,ceci}@di.uniba.it

Abstract. Bisociations represent interesting relationships between seemingly unconnected concepts from two or more contexts. Most of the existing approaches that permit the discovery of bisociations from data rely on the assumption that contexts are static or considered as unchangeable domains. Actually, several real-world domains are intrinsically dynamic and can change over time. The same domain can change and can become completely different from what/how it was before: a dynamic domain observed at different time-points can present different representations and can be reasonably assimilated to a series of distinct static domains. In this work, we investigate the task of linking concepts from a dynamic domain through the discovery of bisociations which link concepts over time. This provides us with a means to unearth linkages which have not been discovered when observing the domain as static, but which may have developed over time, when considering the dynamic nature. We propose a computational solution which, assuming a time interval-based discretization of the domain, explores the spaces of association rules mined in the intervals and chains the rules on the basis of the concept generalization and information theory criteria. The application to the literature-based discovery shows how the method can re-discover known connections in biomedical terminology. Experiments and comparisons using alternative techniques highlight the additional peculiarities of this work.

1 Introduction

Data produced in real-world applications have become so complex, heterogeneous and time-varying that humans are overwhelmed when they attempt to conduct any analysis without technological help. Sophisticated software systems and, in particular, advanced Data Mining techniques are being continuously developed in order to support the analysis of such data and help the comprehension of the underlying phenomena. One of the Data Mining tasks well established but continuously studied is that of association discovery. Typically, associations are based on the notions of co-occurrence, inference of co-occurrence, correlation or similarity which often permit the extraction of useful and interesting connections, but which sometimes represent information already known to the user.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 358–373, 2011.

Especially in the field of science, scientists need to create hypotheses worthy of being investigated and discover connections seemingly remote but supported by an intricate reasoning. Indeed, they have to handle large quantities of data of different natures, intrinsically complex and very often observed in dynamic processes whose structure, components and representation change over time. A part of this problem is elegantly addressed by the approaches which investigate the task of bisociation discovery [1],[13],[9]. By refining the original notion provided in [7], a bisociation is widely recognized as a link that connects concepts from two or more contexts, which are unconnected according to the specific view (very often corresponding to a subjective perspective) by which the contexts are defined. Contexts can be considered as distinct domains which collect a set of concepts, while the discovery of bisociations corresponds to an explorative process which crosses various domains and links concepts present in such domains.

To perform this discovery process two issues have to be addressed[1]: the representation or modeling of the domains and the strategy used to explore the modeled domains and to identify adequate concepts for the linkage. First, the existing approaches exploit a network-based representation which permits the aggregation of various domains (each of which associated to a sub-network) and relates concepts in the same domain and in different domains. The nodes are assigned to the concepts, while the edges express the relationships among the concepts directly observed in the domain or computationally derived, such as the relations of similarity, co-occurrence or probabilistic dependence. Second, interactive navigation techniques and graph analysis algorithms are used to explore the overall network and identify paths which, crossing several sub-networks (distinct domains), link nodes (or other sub-networks) which are far apart in the network and express valuable implicit relations.

Most of the approaches which implement this process rely on the assumption that domains are static, that is, disregard the dynamic nature of several real-world domains and solve only a part of the initial problem of the scientific discovery. Indeed, the network-based representation introduced above models a set of heterogeneous but unchangeable domains, while even a single domain can show changes over time. Indeed, a dynamic domain observed at different timepoints can present different representations and can be reasonably assimilated in a series of distinct static domains. Hence, the necessity to investigate the problem of bisociation discovery arises when taking into account the temporal component and the intrinsic time-varying nature of some domains. This is also justified by the fact that temporal dynamics is attracting interest in the recent data mining literature, since it can play an important role in the comprehension of the evolution of domains under investigation. Among the most significant works, Bottcher et al[2] propose a paradigm based on the temporal dynamics to detect and quantify changes in time-varying models and patterns, while Kleinberg[6] investigates possible approaches to analyze stream-based data from a perspective which considers the temporal evolution of the information.

In this paper, we investigate the task of linking concepts from a dynamic domain through the discovery of bisociations which link concepts over time.

Bisociations permit the representation of linkages which may be unearthed only when considering the dynamic nature and which can not be discovered when considering the domain as static. The two issues (previously described) to perform the process of bisociation discovery are addressed in this work as follows. First, a time interval-based discretization is produced on the domain, so that two time intervals have two different representations of the same domain and therefore, they present somehow two different static domains. Searching for linkages in different domains, even if the data are modeled with the usual network-based representation, can raise computational problems, since the discovery process should evaluate possible links at the level of the original data. To overcome this issue we propose representing each static domain (corresponding to a time-interval) with an abstract description which permits us to focus on the main characteristics of the domain in that time-interval, hence the solution for the other issue. Second, an explorative process across the time-intervals performs the discovery of bisociations among concepts by chaining the abstract descriptions, which involve those concepts, on the basis of the concept generalization and information theory criteria. This allows us to avoid meaningless bisociations and to limit computational problems due to the exploration in the space of the abstract descriptions.

The paper is structured as follows. In the next section we report works related to ours and highlight some peculiarities of the current approach. In Section 3 we formalize the scientific problem studied in this work and in Section 4 we describe the computational solution we propose. The approach is tested with the application to literature-based discovery. Finally, conclusions are drawn.

2 Related Works and Contribution

Current research on bisociations focuses mainly on the discovery of unexpected links from heterogeneous domains by merging conceptual categories. In [9] the authors explore this problem in the analysis of microarray data by proposing a composite framework: the creation of a network-based representation with the integration of different biological repositories and ontologies, grouping of differentially expressed genes (concepts) with a subgroup discovery approach, and discovery of links among the genes contained in the groups. Links are discovered according to a probabilistic approach. A network derived from heterogeneous data sources is also realized in [1] where the authors implement a different discovery approach. In this approach nodes of the networks are assigned to annotated units of information (keywords, gene names), while the edges are weighted to express the degree of certainty and specificity of the relations in the domain between two nodes. Bisociations are discovered with a spreading activation algorithm which is able to extract subnetworks consisting of the most relevant nodes related to a specified set of initially activated nodes. In [13] the problem is explored in document collections, where the application of text-processing techniques permits to obtain a network: the nodes correspond to named entities annotated with a term-frequency vector while the edges are constructed on the basis of the co-occurrence of the entities in the documents. Bisociations are finally discovered by evaluating

the similarity among nodes with vector-based similarity measures. A similar representation is used in [5], where a method combining frequent itemset mining and link analysis is proposed to identify chains of named entities and verbal forms (concepts) extracted from texts. A graph-structure is created by assigning frequent 2-itemsets (pairs of concepts) to a pair of nodes connected by an edge. Final chains of concepts are obtained by following walking paths with an interactive technique which uses statistical measures.

Finding links between seemingly unrelated concepts from a text is a research line started by the pioneering work of Swanson [14] and continued by several approaches of biomedical literature-based discovery. The blueprint of these methods is the A-B-C model [14] where two concepts A and C are given and the discovery process aims to identify the intermediate concept B. Typically, two disjointed literature sets are separately analyzed to mine connections like $A \Rightarrow B$, $B \Rightarrow C$, based on similarity, co-occurrence or correlation. The application of the transitive law would allow us to derive novel connections $A \Rightarrow C$. To obtain the connections $A \Rightarrow B$, $B \Rightarrow C$ two possible strategies can be identified in the literature: *closed discovery*, where the concepts A and C are provided by the user, and *open discovery* in which only A is given. Approaches working on the first strategy have focused mainly on the automatic tools to select the intermediate B concept, for instance in [4] connections are extracted in the form of association rules and the possible intermediate concepts are identified with the integration of domain ontologies. For the second strategy, particular attention is paid to the pruning techniques which eliminate meaningless connections $B \Rightarrow C$. For instance, in [10] the idea is that of considering terms B with respect to a term-frequency based measure, named *rarity*, while in [11] the authors propose knowledge-based heuristics to provide a ranking of the connections $B \Rightarrow C$.

Therefore, linking concepts over time seems to be a not yet investigated issue that would facilitate the discovery of connections between concepts only through a linking process over time. It is noteworthy that the problem here investigated is not different from the bisociation discovery seen in [1],[13],[9] where bisociations connect concepts from unconnected domains identified according to some view of data. In this work, we use the view inherently introduced by time. This means that each domain corresponds to a sort of snapshot of the dynamic domain. The representation of the data is another distinguishing aspect of the current approach from the others. Indeed, the domain is modeled with abstract representations in the form of lattice-based structures of multiple level association rules. Since rules denote statistical evidence, the usage of abstract descriptions justifies the robustness of the method by reducing the risk of false positive links.

3 Formal Definition of the Problem

Before formally defining the scientific problem of interest in this work, here we introduce some preliminary concepts. Let $O_D : \langle O_1, O_2 \dots O_i \dots O_n \rangle$ be a sequence of time-ordered observations on the concepts $\mathcal{C}$ of the domain D. For instance, in the domain of biomedical literature the set $\mathcal{C}$ would correspond to a set of biomedical named entities, while each observation O_i is assigned to a

single paper with a specific publication date. Therefore, a subset of the named entities $\mathcal{C}$ is observed in a paper O_i.

Given a language $\mathcal{L}$ defined on the concepts $\mathcal{C}$, let $\mathcal{A}$ be a set of statements in $\mathcal{L}$ produced by applying an operator $\mathcal{M}$ to a subsequence $O_i \dots O_j$ of O_D $(i < j)$. $\mathcal{M}$ provides abstract descriptions of subsequences $O_i \dots O_j$ $(i, j = 1 \dots n)$. Each abstract description can be denoted with statistical parameters or certainty measures. By following the example above, $\mathcal{M}$ would generate frequent patterns $\mathcal{A}$ from the named entities in the set of papers $O_i \dots O_j$.

Let $\mathcal{T}$ be an operator which maps the set of time-stamps $\{t_1 \dots t_n\}$ of $\langle O_1, O_2 \dots O_i \dots O_n \rangle$ into τ, where $\tau : \{\tau_1, \dots \tau_n\}$ is a finite totally ordered set of time-points under the order relation denoted by " $\leq$ ". $\mathcal{T}$ permits to discretize time-stamps such that, given two time-stamps t_i, t_j for which $t_i < t_j$, also $\mathcal{T}(t_i) \leq \mathcal{T}(t_j)$. For instance, given three publication dates "April 20 2010", "May 10 2010", "May 10 2011": $\mathcal{T}$(April 20 2010)= $\mathcal{T}$(May 10 2010)= 2010, $\mathcal{T}$(May 10 2011)= 2011. Therefore, we can associate the sequence $O_D : \langle O_1, O_2 \dots O_i, O_{i+1}, \dots O_{n-1}, O_n \rangle$ of time-stamped observations with a sequence $\{\tau_1, \tau_2, \dots, \tau_i, \tau_{i+1}, \dots \tau_m\}$, $(\tau_1 < \tau_2 < \dots < \tau_m)$. For instance, given $O_D : \langle April$ 1 2008, $April$ 2 2008 $\dots$ May 1 2008, May 2 2008, $\dots$ $April$ 1 2009, $April$ 2 2009, $\dots$ $April$ 1 2010, $April$ 2 2010, $\dots$ $April$ 1 2011, $April$ 2 2011, $\rangle$, we can associate it with the sequence $\{2008, 2009, 2010, 2011\}$.

Definition 1. *Given X, Y concepts in $\mathcal{C}$, a temporal bisociation B is a sequence of abstract descriptions $A_1, A_2, \dots A_{m-1}$, $A_1 \in \mathcal{A}_1$, $A_2 \in \mathcal{A}_2, \dots A_{m-1} \in \mathcal{A}_{m-1}$, where $\mathcal{A}_i$ is obtained from the observations included in $[\tau_i; \tau_{i+1}]$. A_1, A_{m-1} involve the target concepts X, Y respectively.*

Informally, Definition 1 states that, given a sequence of time-intervals $[\tau_1; \tau_2]$, $[\tau_2; \tau_3]$,... $[\tau_{m-1}; \tau_m]$, the abstract descriptions $\mathcal{A}_1, \mathcal{A}_2 \dots \mathcal{A}_{m-1}$ can be derived from them. The sequence $A_1, A_2, \dots A_{m-1}$ reports the bisociation from X to Y. Without loss of generality, in this work abstract descriptions are computed in the form of association rules so, for instance, the chain of rules $X \Rightarrow \mathbf{W}$, $\mathbf{W} \Rightarrow \mathbf{J}$, $\mathbf{J} \Rightarrow \mathbf{Z}$, $\mathbf{Z} \Rightarrow Y$ (**W,J,Z** sets of intermediate concepts) stands for a bisociation linking X to Y ($X \Rightarrow \mathbf{W}$, $\mathbf{W} \Rightarrow \mathbf{J}$, $\mathbf{J} \Rightarrow \mathbf{Z}$, $\mathbf{Z} \Rightarrow Y$ mined from the observations associated to four distinct consecutive time-intervals).

Considering the notions introduced so far, the problem of discovering temporal bisociations can be divided into two sub-problems:

1. *Given*: $O_D : \langle O_1, O_2 \dots O_i \dots O_n \rangle$, $\mathcal{T}$ such that the width of each time-interval $[\tau_r; \tau_{r+1}]$ is greater than or equal to a user-defined threshold $\Delta_{\mathcal{T}}$, a certainty measure C, *Find*: a set $R_{\mathcal{A}} : \{\mathcal{A}_1, \mathcal{A}_2, \dots \mathcal{A}_{m-1}\}$ of abstract representations satisfying the certainty measure C.
2. *Given*: two concepts $X, Y \in \mathcal{C}$, the set $R_{\mathcal{A}}$, a minimum number $\eta_{\mathcal{T}}$ of time-intervals to be crossed, a certainty measure M, *Find*: a collection $\mathcal{B}$ of temporal bisociations $A_1, A_2, \dots A_{m-1}$ meeting the certainty measure M with $(m-1) \geq \eta_{\mathcal{T}}$.

A computational solution to these sub-problems is described in the following.

4 Discovering Temporal Bisociations

We should remember that solving the two sub-problems previously formalized means addressing respectively the two issues introduced in the Section 1 when discovering bisociations. So, the approach which finds the set $R_{\mathcal{A}}$ actually permits us to define a representation of the domains. While, the approach which uses the set $R_{\mathcal{A}}$ to find the collection $\mathcal{B}$ integrates a domain exploration strategy in order to identify bisociations. The computational solution comprises a preliminary step aiming to exclude the trivial connections between X and Y, namely it checks that the two concepts are not already directly connected or that there is no obvious evidence which connects them.

4.1 Check for Direct Connections

Direct connections among the concepts X and Y can be expressed by either similarities or co-occurrences. In this work, we follow the second way, since the first one would require the usage of (dis)similarity measures, semantics and ontologies which can be cumbersome and computationally expensive in many applications. The check is performed by controlling the absence of statistical evidence of the connections both at the level of the static domains (each time-interval) and at the level of the dynamic domain (cross time-intervals). The technique to determine statistical evidence used in this work is that of association rule mining: rules which present the concepts on either the antecedent side or on the consequent side denote reasonably direct connections between the concepts, since X and Y co-occur in the set of supporting observations O_i. To perform this preliminary step we exploit the algorithm proposed in [8] which enables the discovery of non-redundant association rules. More precisely (Algorithm 1), the algorithm is first applied to the complete set of observations O_D (lines 4-8) and then separately to each partition $P \in \mathcal{P}$ produced by applying the operator $\mathcal{T}$ to O_D (lines 9-16). The width of each partition (time-interval) is forced to be bigger than $\Delta_{\mathcal{T}}$. A description of the algorithm is reported in the next section.

4.2 Generation of Abstract Descriptions

Once possible statistical evidences have been excluded, each of the partitions $\mathcal{P}$ of the observations O_D, produced by the application of $\mathcal{T}$, is represented as abstract representations in the form of association rules (ARs). These reflect the statistical regularities in the data of that partition $P \in \mathcal{P}$ and move the discovery of bisociations at upward to higher abstraction level resulting in a reduction of the risk of false positive links. In these terms, the certainty measure C in the formulated problem (Section 3) consists of the usual statistical parameters *support* and $confidence$ which denote the rules, so the resulting abstract descriptions are ARs, which meet the minimum thresholds of support and confidence. By following this idea, we integrate into the process of AR mining ontologies of the dynamic domain, which permit us to annotate or abstract the intermediate concepts of the links. Exploiting domain ontologies (or more generally, background

Algorithm 1. Check of direct connections.

```
1: input: O_D, X, Y, ARM, minSup, minConf, T, Δ_T                output: CHECK
   // ARM algorithm of association rules mining
2: CHECK := false
3: AR ← ARM(O_D)  // AR association rules mined from the data O_D
4: for all R ∈ AR do
5:    if X ∈ R and Y ∈ R then
6:       CHECK := true
7:    end if
8: end for
9: P := partitioning(O_D, T, Δ_T)
10: for all P ∈ P do
11:    AR ← ARM(P)  // AR association rules mined from the data P
12:    for all R ∈ AR do
13:       if X ∈ R and Y ∈ R then
14:          CHECK := true
15:       end if
16:    end for
17: end for
```

knowledge) on the concepts is not actually a novelty in bisociation discovery: in [9] the authors use biological ontologies to annotate gene sets, while, in this work, we resort to background $is-a$ hierarchies, which generalize the concepts and which exploit the is-a relationships among the occurring concepts to strengthen the reliability of links. The process of AR mining is performed by means of the algorithm proposed in [8] which enables the discovery of non-redundant ARs at several hierarchical levels from data represented in the quite simple form of attribute-value pairs. The possibility of pruning redundancies of that algorithm here turns out to be an important peculiarity which makes the resulting abstract descriptions compact and without superfluous information. We report a brief description of the algorithm of ARs mining in the following.

The algorithm is composed of two steps which permit respectively to i) generate the set of closed frequent itemsets (in this work, sets of concepts) at the different hierarchical levels whose support exceeds the minimum threshold and ii) discover from these itemsets ARs at the different hierarchical levels (multiple level ARs) whose confidence exceeds the minimum threshold. The first step implements the notion of *closed* itemsets when the items are hierarchically organized. Mining the closed itemsets from a set D of observations means mining the maximal elements of the equivalence classes of the all itemsets derived from D[1]. Hence, an itemset Y=$\langle$ y_1 , y_2 ,..., y_j ,...,$y_h\rangle$ is closed iff no supersets of Y is supported by the same set of observations of Y, and therefore, by the same support of Y. When itemsets can be organized according to an is-a hierarchy H, the concept of closed itemset has to be extended to that of **multiple level closed itemset** to remove redundant (according to H) multiple level itemsets.

The algorithm proceeds by scanning the hierarchy in top-down mode while, at each level, it generates a set of multiple-level closed itemsets with increasing length. The length is the number of items present in an itemset. The second step

[1] Two itemsets belong to the same equivalence class when they cover the same observations.

extends the notion of *minimality* to the ARs derived by the itemsets produced in the first step. Formally speaking:

Definition 2. *An association rule* $R_1 : A_1 \Rightarrow C_1$ *is* ***minimal*** *iff* $\nexists R_2 : A_2 \Rightarrow C_2$ *with identical support and confidence of* R_1*, for which* $A_2 \subseteq A_1$*,* $C_1 \subseteq C_2$*.*

A interpretation of this definition for this work is that the minimal rules convey additional inferential information by means of the inclusion relationships of the antecedents and consequents of the rules. Indeed, if we consider $R_1 : A \Rightarrow B, C$, $R_2 : A, B \Rightarrow C$, with identical support and confidence, the antecedent of R_1 is included in the antecedent of R_2 while the consequent of R_1 includes the consequent of R_2. Hence R_1 is minimal with respect to R_2 and gives more information on the consequent side than R_2. R_2 is considered redundant.

4.3 Linking Concepts over Time

Once the ARs are discovered in each partition P on O_D, they are organized in a lattice-based structure $(R_{\mathcal{A}} : \{\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_{m-1}\})$: the nodes of the lattice represent ARs, while the edges represent relationships between the ARs. A finite sequence of edges which relate two ARs is called a *path*. Three types of path originate from the root of a lattice: a) paths for the generalization of the concepts contained in the root, b) paths for the extension of the rule at the root with larger rules, c) paths for the generalization of the concepts contained in the root with larger rules. In Figure 1b, $A_{11}, B_{12} \Rightarrow B_{11}$ is a node of the paths of type a (arrow), $A_1 \Rightarrow B_{11}$ is a node of the paths of type b (thick arrow), according to the hierarchy in Figure 1a, while $A_1, B_{12} \Rightarrow B_{11}$ is a node of the paths of type c (double arrow) according with the same hierarchy. In these three cases the rules are positioned in the lattice by child-father relationships of the hierarchy H and increasing length: that is, the rules at level $k+1$ contain father concepts of the concepts contained in the rules at level k and present one more concept than the rules at the level k. This permits an early evaluation of the rules which contain a low number of concepts among those occurring in the observations. Moreover, integrating heuristics on the paths and the organization of the nodes permits us to conduct an informed search in the lattices thus reducing the overall computational cost.

The value of a certainty measure M is associated to each rule: in this work mutual information plays the role of M introduced in the problem formulation (Section 3). Mutual information (mi) is one of the quantitative measures which can denote a rule and it has the peculiarity to express the mutual dependence between the antecedent and the consequent of a rule. What is more interesting is that it represents the ratio of the actual probability of two concepts to be related to the probability of two concepts to be unrelated. In this work, we prefer mi to other typical parameters, such as confidence. Actually, the confidence is not an appropriate measure of correlation strength between concepts since it leads to select common concepts in the consequents and rare concepts in the antecedents. The consequents in these cases rarely add much meaning to the final link. Differently, mutual information emphasizes relatively rare concepts that generally

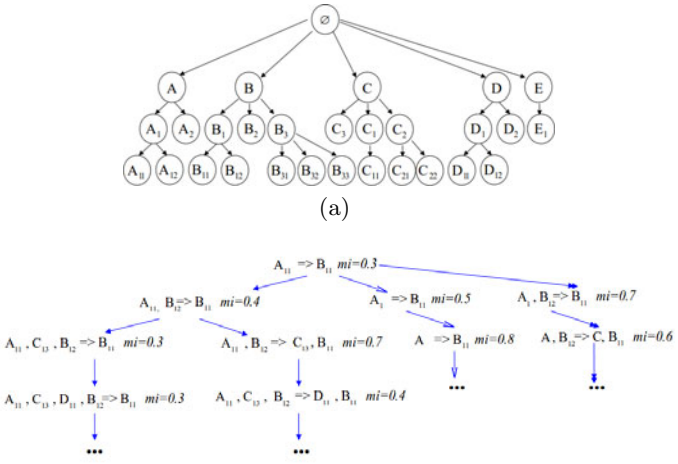


Fig. 1. Is-a hierarchy over the concepts and a portion of the lattice of multiple-level association rules produced from a partition with their mi values

occur together and mitigates the importance of common concepts, thus leading to the discovery of more interesting bisociations. Mutual information is computed as $log\frac{support(Antecedent,Consequent)}{support(Antecedent)*support(Consequent)}$. ARs with mutual information less than a user-defined minimum threshold σ_{mi} are not considered.

The discovery of temporal bisociations is performed through an explorative process which crosses the lattices of the time-intervals and chains the included ARs by considering two different, complementary directions. The semantics associated to the concepts, expressed in the form of concept generalization and the information theory measure, in form of the mutual information are associated to each AR. Chaining is the basic operation which produces the link between two ARs: R_1, R_2 discovered in two consecutive time-intervals. Links are produced when the antecedent of R_2 contains either concepts present in the consequent of R_1 or more general concepts than those present in the consequent of R_1. A sequence of chained ARs constitutes a temporal bisociation: final bisociations have to include a number of ARs greater than or equal to a user-defined threshold $\eta_{\mathcal{T}}$. In other words, we are interested in temporal bisociations which have been developed over at least $\eta_{\mathcal{T}}$ consecutive time-intervals and which therefore involve at least $\eta_{\mathcal{T}}$ concepts. In particular, the root of the first lattice involved in a bisociation ($\mathcal{A}_1$) contains a rule (A_1), whose antecedent presents only the target concepts X, while the last lattice involved in the same bisociation ($\mathcal{A}_{m-1}$) contains a rule (A_1), whose consequent presents only the target concept Y. The total number of lattices to be explored has to be greater than or equal to $\eta_{\mathcal{T}}$: bisociations discovered from non-consecutive lattices or which do not cover this time span will be not considered.

The explorative process integrates a depth-first search by visiting the paths in this order: type a, b, c. In each lattice, the exploration visits the nodes by starting from the root: if the value of mi of the current node exceeds σ_{mi}, then the exploration in that lattice is completed and the antecedent of the rule in the current node is used as a "bridge" to explore the lattice of the next time-interval to link one of the contained rules. Otherwise, the search continues downward level-by-level by considering nodes of the same type of path until it reaches the leaf nodes. Then, it proceeds by backtracking and continues to explore paths of the same type or, when the paths of the same type have been completed, it goes back to the root and continues on paths of another type.

The linking process associated to the lattice of the next time-interval searches for a rule suitable for the linkage according to the following modalities: 1) rules of length two (one concept in the antecedent-one concept in the consequent), whose antecedents contain only the consequent of the final rule of the previous lattice; 2) rules of length greater than two, whose antecedents contain also the consequent of the final rule of the previous lattice; 3)rules of length two, whose antecedents contain only one concept which generalizes (according to the hierarchy H) the consequent of the final rule of the previous lattice; 4) rules of length greater than two, whose antecedents contain also a concept which generalizes the consequent of the final rule of the previous lattice. When several rules are identified as possible roots of the lattice to be explored, then the rule with higher mi value is selected. The discovery process continues by combining the exploration in each lattice (previously described) and the linking technique between lattices of the consecutive time-intervals up to the last lattice which completes the bisociation with a rule, whose consequent will be the target concept Y.

A trace of the discovery process is reported in Figure 3. Consider the time-intervals [1990;1992],[1992;1994], [1994;1996], σ_{mi}=0.5 and let A_{11} be the target object X. The process starts by searching in the lattice of [1990;1992] for the "entry" rule for the exploration, namely a rule whose antecedent is A_{11} (Figure 3a). Once identified, the exploration proceeds by evaluating mi of the rules with paths of type a: first the branch annotated with square 1, where no rule exceeding σ_{mi} is found, then the branch with square 2, where we have the same result. Subsequently, the path of type b (square 3) is explored, where the rule $A_{11} \Rightarrow B_{11}$ exceeds σ_{mi} (bold square 3): therefore the concept B_{11} becomes the bridge between the time-intervals [1990;1992],[1992;1994].

The exploration in [1992;1994] starts by searching for rules (with higher mi) whose antecedents contain B_{11} (Figure 3a). Once the "entry" rule has been identified (modality 3 above), the nodes of the branches with square 3, 4, 5 (types a, c) are evaluated, but none of them with success. Once the lattice has been completely explored, a new root is identified as the rule containing B_{11} on the antecedent (modality 3) and a new exploration in a new lattice of the same time-interval starts (Figure 3b). By following the same exploration strategy, the rule which continues the linkage is $A_{12}, C_{13}, B_1 \Rightarrow C, D_{11}$ (bold square 4). Hence, the list of concepts C, D_{11} becomes the bridge between the time-intervals [1992;1994],[1994;1996], as a further contribution to the bisociation. The

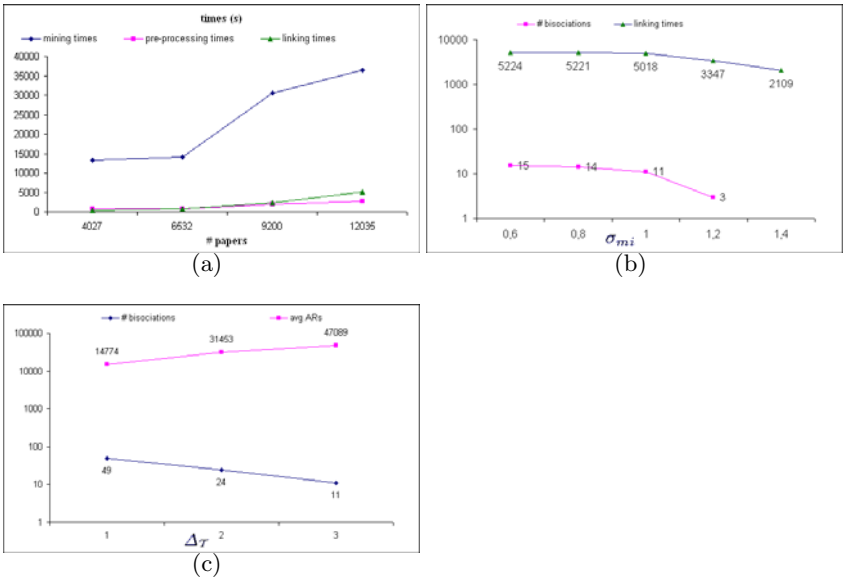


Fig. 2. Running times as a function of the number of papers published in [2000;2009] ($\Delta_T = 1$) (a). Number of bisociations as a function of σ_{mi} (b) and Δ_T (c).

exploration in [1994;1996] starts by searching for the rule (with higher mi), whose antecedents contain one of the possible permutations of C, D_{11} (Figure 3c). The identified root is the rule $C, D_{11} \Rightarrow E_1$ (modality 2), which provides also the bridge E_1 for the next interval.

Note that strategies to improve the exploration of the lattices may turn out to be ineffective in the case of informed searches. Moreover, pruning techniques would be inapplicable considering that, for the measure of mutual information, the *anti-monotonic* property does not hold for either rules with generalized concepts or rules with different length or with identical length.

5 Experiments on Biomedical Literature

One of the widely recognized dynamic domains is the scientific literature. It represents the typical source of information which researchers exploit for their studies and the typical means to disseminate their investigations. Publications may report studies on the same topic conducted one after another over time and this motivates our vision of the scientific literature as a dynamic domain. Literature is therefore a natural field to prove the viability of automatic tools for scientific discovery. For the current work, biomedical literature also represents the board on which to compare existing techniques with the one we are proposing. In this sense, these experiments aim at re-discovering known connections in biomedical terminology and highlighting potentialities offered by this work.

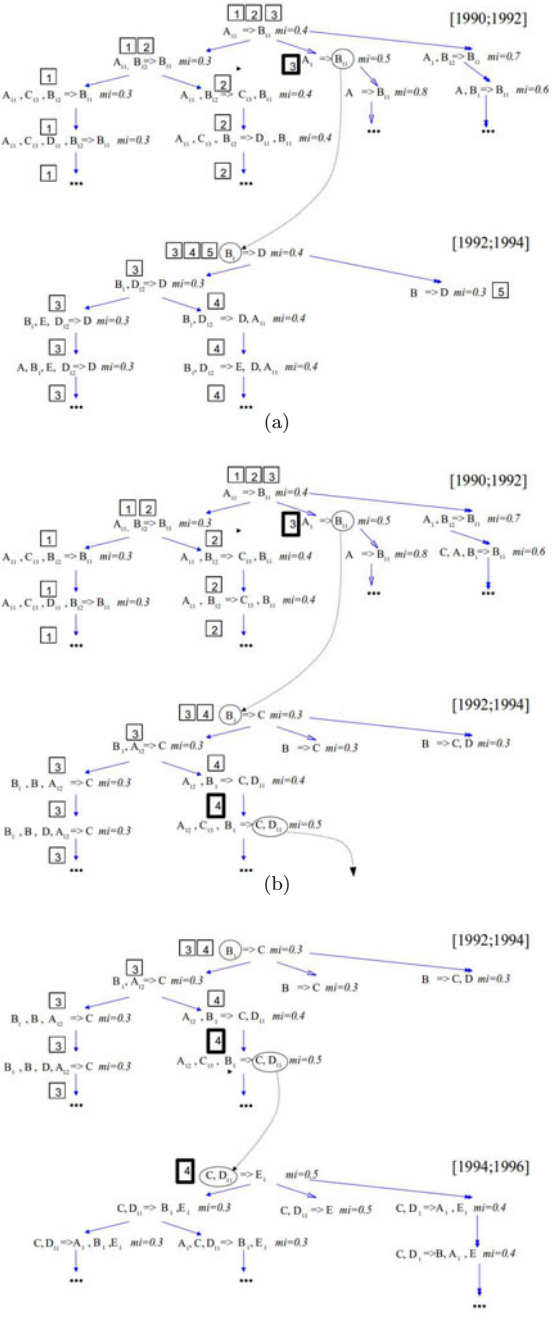


Fig. 3. Linking concepts over three time-intervals

Experimental Setup. Dealing with publications requires a necessary pre-processing step which permits the generation of the set of observations O_D. The original data set was composed of publications retrieved by the Pubmed search engine. In particular, since we tested our approach on the biomedical literature-based Swanson discoveries [14], the collection of original publications was generated from the result sets returned by Pubmed when, in December 2009, we submitted two distinct queries, namely "migraine", "magnesium deficiency". The two publication sets were obtained amounting to 5311 and 22223, respectively. The title, publication date and abstract sections were considered and pre-processed. Basic natural language processing techniques available in the GATE framework[2] were applied in order to identify biomedical named entities. This result was also obtained integrating controlled domain thesauri, such as MEsh Terms vocabulary[3], whose taxonomic organization allowed the production of the hierarchy H used for the generation of abstract descriptions (Section 4.2). The terminology available in the thesauri produces the set C of concepts.

The problem of the presence of synonyms was also addressed by integrating linguistic resources which permit the replacement of variant names of biomedical entities with their canonical names. The application of the operator $\mathcal{T}$ enables the discretization over time of the complete dynamics of the biomedical literature into partitions of publications published in time-intervals, based on the temporal dimension of the years. Each pre-processed paper corresponds to an observation O_i included in a time-interval and it can be interpreted as the investigation of the scientists at a specific time-stamp. The generation of the multiple-level ARs (abstract descriptions) was performed on sets of data, each of which is composed of the subset of pre-processed papers included in a time-interval. A further pre-processing was conducted by selecting the subset of concepts occurring in each paper whose TF-IDF measure[12] exceeded a user-defined minimum threshold.

Experiments were performed considering four different criteria, namely scalability, influence of the input parameters on the temporal bisociations patterns, information conveyed in the bisociations and comparison with existing solutions.

Scalability. Experiments on the performances in time were performed when increasing the threshold $\eta_{\mathcal{T}}$ and hence the number of papers, while the value of $\Delta_{\mathcal{T}}$ is 1 year (width of the time-intervals), $minSup$=0.3, $minConf$=0.7, σ_{mi}=1, minimum TF-IDF = 0.3. Collected running times consider the step of pre-processing, the AR mining algorithm and the discovery of temporal bisociations. In Figure 2a the results obtained considering the whole set of papers published in [2000;2009] are reported. They show that the computational cost is mainly due to the step of abstract description generation, which, however, returns a number of rules multiplied by a factor 3, while the number of papers increases of the same factor (from 4027 to 12035). This is also justified by the fact that the mining algorithm generates rules at different hierarchical levels, given that it integrates the hierarchy organizing the concepts. Indeed, when # papers is 12035 we have the highest number of ARs and the highest average of papers per time-interval

[2] http://gate.ac.uk/family/
[3] http://www.nlm.nih.gov/mesh/

(Table 1). The running times of the linking process are encouraging since it increases linearly with respect to the number of considered time-intervals (and therefore number lattices to be visited). This performance is due to the used heuristics which avoid a greedy exploration of the lattices.

Influence of Parameters. We tested the proposed computational solution when tuning the minimum threshold σ_{mi} and $\eta_{\mathcal{T}}$ ($minSup$=0.3, $minConf$=0.7). A initial consideration can be drawn from the results in Figure 2b (performed for the papers published in [1990;2008], $\eta_{\mathcal{T}}$=9, $\Delta_{\mathcal{T}}$=3 years), which empirically confirm the influence of the mutual information on the bisociations. It emerges that the most discriminative values of mi are basically included in the range (1;1.4], therefore, tuning σ_{mi} to values lower than 1 does not require additional computational cost and leads to the discovery of approximately the same set of bisociations. In fact, this is due to the replacement of variant names of concepts with canonical names, whose co-occurrences tend to strengthen their dependence. An interesting aspect is the generation of a maintainable set of bisociations which the user can easily investigate. This is due to fact that the setting requires the discovery of bisociations linking concepts over a relatively wide period of 27 years, namely 9 time-intervals, each of which covers 3 years. The experiments in Figure 2c (σ_{mi}=1, $\eta_{\mathcal{T}}$=9), on the influence of $\Delta_{\mathcal{T}}$, confirm the maintainability of the discovered bisociations, especially when compared to the number of rules (at the worst, 49 against 14774). Indeed, when $\Delta_{\mathcal{T}}$=1 the maximum number of lattices to be explored is generated, which generally leads to the strong increase of bisociations and to the reduction of the average number of ARs per lattice.

Comparison with Existing Techniques. The approach was compared with the existing systems *BITOLA*[3] and *ARROWSMITH*[15] focusing on the problem of literature-based discovery and on the same original data. These systems work in an interactive way, so comparing running times does not give any indications. By submitting the concepts X as "magnesium deficiency" and Z "migraine", *BITOLA* discovers 2620 possible linking concepts Y able to form links with three concepts. For each pair (X, Y), (Y, Z), statistical parameters (e.g., frequency) are provided, although the huge set of intermediate concepts could be cumbersome for the user. On the other hand, our approach, by setting the target concepts X as "magnesium deficiency" and Y as "migraine", discovers only one temporal bisociation (even tuning σ_{mi} in [0,5;1]). By setting $minSup$=0.3, $minConf$=0.4, $\Delta_{\mathcal{T}}$= 1 year, $\eta_{\mathcal{T}}$=2 the following bisociation between "magnesium deficiency" and "migraine" is discovered in [1990; 1997] (the contribution of each intermediate concept is temporally collocated):

Table 1. Total and average abstract descriptions by increasing the number of papers

$[\eta_{\mathcal{T}}]$	$[\tau_1;\tau_m]$	# papers	# ARs	avg ARs
3	[2000;2003]	1342	15302	4027
5	[2000;2005]	2177	16143	6532
7	[2000;2007]	3066	22880	9200
9	[2000;2009]	4011	27908	12035

[1990;1991] Magnesium deficiency AND Anatomy ⇒ Metals, alkaline earth AND Metals [support=0.30, confidence=1.0, mi=1.19]

[1991;1992] Metals, alkaline earth AND Metals AND Diseases ⇒ Metals, light [support=0.33, confidence=0.97, mi=1.075]

[1992;1993] Chemicals and drugs AND Neurologic manifestations ⇒ Signs and symptoms [support=0.31, confidence=1.0, mi= 1.17]

[1993;1994] Central Nervous System Diseases AND Signs and Symptoms ⇒ Neurologic Manifestations [support=0.3, confidence=0.97, mi=1.178]

[1994;1995] Biological Sciences AND Neurologic Manifestations ⇒ Pathological Conditions, Signs and Symptoms [support=0.318, confidence=1.0, mi=1.144]

[1995;1997] Headache Disorders AND Pathological Conditions, Signs and Symptoms ⇒ Migraine [support=0.30, confidence=1.0, mi=1.185]

An identical comparison was performed with *ARROWSMITH*, which, however, requires more human intervention to carry out the process. The search for intermediate concepts from "magnesium deficiency" to "migraine" produces 598 possible links which can be further investigated with the support of the user, while the proposed solution requires less interaction.

We also evaluated the final bisociations by considering Swanson's linking terms [14] as gold standard. In his work, eleven hidden connections between "magnesium deficiency" to "migraine" were identified with the A-B-C model, whose intermediate concepts were: *Type A personality, vascular reactivity, calcium blockers, platelet activity, spreading depression, epilepsy, serotonin, inflammation, prostaglandins, substance P, brain hypoxia.* A subset of four terms, namely: epilepsy, serotonin, inflammation, prostaglandins, were contained in the controlled vocabulary and is-a hierarchy that we used, while the others were not recognized. Temporal bisociations discovered in [1989;1997] (period of investigation in [14]) involved effectively those four concepts and more general concepts (father concepts) than the former according to the hierarchy H.

Another consideration can be made from a basic aspect of the approach: the process of linking X to Y can produce different bisociations from linking Y to X over time, therefore, the temporal order is relevant in this work and permits us to determine the collocation over time of each contributing intermediate concept. For instance, the set of bisociations obtained from [1980;2009] ($minSup$=0.3, $minConf$ =0.4, σ_{mi}=0.8, $\Delta_{\mathcal{T}}$= 2 years, $\eta_{\mathcal{T}}$=2, X= "magnesium deficiency", Y="migraine") amounts to only one (not reported here due to lack of space) which involves thirteen concepts. If X="migraine", Y= "magnesium deficiency" one bisociation which involves seven intermediate concepts is discovered.

6 Conclusions

We have presented a novel approach to discover bisociations when considering the time-varying nature of the domains. Contrary to previous approaches, the discovery is performed on abstract descriptions of the domains which provide several advantages: focus on the main characteristics of the domains, prevention of computational cost due to the search in the original data and the reduction of the risk of false positive links. The linking process exploits two criteria, one based on semantics, the other one based on information theory. The application to the problem of literature-based discovery proves the reproducibility of the known

results and the scalability of the approach. For future work, we plan to further improve the search in the lattices and extend experiments to other scenarios.

Acknowledgment. This work is supported by the research project ATENEO-2010: "Modelli e Metodi Computazionali per la Scoperta di Conoscenza in Dati Spazio-Temporali".

References

1. Berthold, M.R., Dill, F., Kötter, T., Thiel, K.: Supporting creativity: Towards associative discovery of new insights. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 14–25. Springer, Heidelberg (2008)
2. Böttcher, M., Höppner, F., Spiliopoulou, M.: On exploiting the power of time in data mining. SIGKDD Explorations 10(2), 3–11 (2008)
3. Hristovski, D., Peterlin, B., Mitchell, J.A., Humphrey, S.M.: Using literature-based discovery to identify disease candidate genes. I. J. Med. Inf. 74(2-4) (2005)
4. Hu, X., Zhang, X., Yoo, I., Wang, X., Feng, J.: Mining hidden connections among biomedical concepts from disjoint biomedical literature sets through semantic-based association rule. Int. J. Intell. Syst. 25(2), 207–223 (2010)
5. Jin, W., Srihari, R.K., Ho, H.H., Wu, X.: Improving knowledge discovery in document collections through combining text retrieval and link analysis techniques. In: ICDM 2007, pp. 193–202 (2007)
6. Kleinberg, J.: Temporal dynamics of on-line information streams. Data Stream Management: Processing High-Speed Data Streams (2006)
7. Koestler, A.: The act of Creation. London Hutchinson (1964)
8. Loglisci, C., Malerba, D.: Mining multiple level non-redundant association rules through two-fold pruning of redundancies. In: Perner, P. (ed.) MLDM 2009. LNCS, vol. 5632, pp. 251–265. Springer, Heidelberg (2009)
9. Mozetic, I., Lavrac, N., Podpecan, V., Novak, P.K., Motain, H., Petek, M., Gruden, K., Toivonen, H., Kulovesi, K.: Bisociative knowledge discovery for microarray data analysis. In: 1st Int. Conf. on Computational Creativity, Lisbon, Portugal (2010)
10. Petric, I., Urbancic, T., Cestnik, B., Macedoni-Luksic, M.: Literature mining method rajolink for uncovering relations between biomedical concepts. Jour. of Biom. Inf. 42(2), 219–227 (2009)
11. Pratt, W., Yetisgen-Yildiz, M.: Litlinker: capturing connections across the biomedical literature. In: Gennari, J.H., Porter, B.W., Gil, Y. (eds.) K-CAP, pp. 105–112. ACM, New York (2003)
12. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill Book Company, New York (1984)
13. Segond, M., Borgelt, C.: Selecting the links in bisoNets generated from document collections. In: Cohen, P.R., Adams, N.M., Berthold, M.R. (eds.) IDA 2010. LNCS, vol. 6065, pp. 196–207. Springer, Heidelberg (2010)
14. Swanson, D.R.: Migraine and magnesium: Eleven neglected connections. Perspectives in Biology and Medicine 31, 526–557 (1988)
15. Swanson, D.R., Smalheiser, N.R.: Implicit text linkages between medline records: Using arrowsmith as aid to scientific discovery. Library Trends 48(1) (1999)

Minimum Neighbor Distance Estimators of Intrinsic Dimension

Gabriele Lombardi, Alessandro Rozza, Claudio Ceruti,
Elena Casiraghi, and Paola Campadelli

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano,
Via Comelico 39-41, 20135 Milano, Italy
lombardi@dsi.unimi.it
http://homes.dsi.unimi.it/~campadel/LAIV/index.htm

Abstract. Most of the machine learning techniques suffer the "curse of dimensionality" effect when applied to high dimensional data. To face this limitation, a common preprocessing step consists in employing a dimensionality reduction technique. In literature, a great deal of research work has been devoted to the development of algorithms performing this task. Often, these techniques require as parameter the number of dimensions to be retained; to this aim, they need to estimate the "intrinsic dimensionality" of the given dataset, which refers to the minimum number of degrees of freedom needed to capture all the information carried by the data. Although many estimation techniques have been proposed, most of them fail in case of noisy data or when the intrinsic dimensionality is too high. In this paper we present a family of estimators based on the probability density function of the normalized nearest neighbor distance. We evaluate the proposed techniques on both synthetic and real datasets comparing their performances with those obtained by state of the art algorithms; the achieved results prove that the proposed methods are promising.

Keywords: Intrinsic dimensionality estimation, dimensionality reduction, manifold learning.

1 Introduction

Most of the machine learning techniques suffer the "curse of dimensionality" (Hughes effect) when applied to high dimensional data; for these reasons, many real life signals characterized by high dimensionality, such as images, genome sequences, or EEG data, cannot be successfully processed. To face this limitation, a first step of dimensionality reduction is needed to project the data onto lower dimensional spaces where they are tractable. This step could be profitable when the reduced data are projected along the most characterizing dimensions. Indeed, most real data can be fully characterized by few degrees of freedom, represented by low dimensional feature vectors; in this case the feature vectors can be viewed as points constrained to lie on a low dimensional manifold embedded

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 374–389, 2011.

in a higher dimensional space. The dimensionality of these manifolds is generally referred as *intrinsic dimensionality* (id). In other words, the id of a given dataset $\boldsymbol{X}_N \equiv \{\boldsymbol{x}_i\}_{i=1}^N \subset \Re^D$ is the minimum number of parameters needed to capture, and describe, all the information carried by the data. In more general terms, according to [11], $\boldsymbol{X}_N$ is said to have id equal to $d \in \{1..D\}$ if its elements lie entirely within a d-dimensional subspace of $\Re^D$.

Given a high dimensional dataset, the estimation of its id would be a fundamental step of any dimensionality reduction technique. Unfortunately, to our knowledge, existing techniques fail dealing with high id non-linear datasets, and the problem is still open.

In this work we present the following intrinsic dimension estimation methods: the "Minimum Neighbor Distance - Maximum Likelihood" estimator and its variants ($\texttt{MiND}_{\texttt{ML*}}$), and the "Minimum Neighbor Distance - Kullback Leibler" estimator ($\texttt{MiND}_{\texttt{KL}}$); moreover, we compare them with state of the art algorithms.

In Section 2 the related works are summarized; in Section 3 our theoretical results and our estimators are proposed; in Section 4 experimental settings and results are reported; in Section 5 conclusions and future works are presented.

2 Related Works

The most cited example of id estimator is the Principal Component Analysis (PCA) [14], which is a well known technique that is often used as the first step of many machine learning methods to reduce the data dimensionality. To this aim, PCA projects points on the directions of their maximum variance, which are estimated by computing the data covariance matrix and performing eigendecomposition. Exploiting PCA, the intrinsic dimension d can be estimated by counting the number of normalized eigenvalues that are higher than a threshold parameter λ. In [10] the author achieves more accurate results by applying PCA in small subregions of the dataset to estimate their local id. The id of the whole dataset is then determined by combining all the local ids. The problem of using PCA relies in the difficulty of choosing a proper value for λ, which strongly affects the accuracy of the estimated id. Moreover, PCA is a linear technique and cannot successfully deal with non linear datasets.

Another well-known id estimator is the Packing Number technique [15]. This method exploits the r-packing number $M(r)$ of the dataset $\boldsymbol{X}_N \subset \boldsymbol{\mathcal{S}}$, where $\boldsymbol{\mathcal{S}}$ is a metric space with distance metric $\delta(\cdot,\cdot)$. More precisely, $\boldsymbol{X}_N$ is said to be r-separated if $\forall\ \boldsymbol{x},\boldsymbol{y} \in \boldsymbol{X}_N, \boldsymbol{x} \neq \boldsymbol{y} \Rightarrow \delta(\boldsymbol{x},\boldsymbol{y}) \geq r$, and $M(r)$ is the maximum cardinality of an r-separated subset of $\boldsymbol{X}_N$. Given this definition, the authors demonstrate that the id of $\boldsymbol{X}_N$ can be found by approximating the limit:

$$d = -\lim_{r \to 0} \frac{\log M(r)}{\log r} \qquad \text{with} \qquad \hat{d} = -\frac{\log(M(r_2) - M(r_1))}{\log(r_2 - r_1)}$$

where $r_2 > r_1$ are two radiuses to be set as parameters.

The previously described approaches often fail when managing non-linearly embedded manifolds or noisy data, and they become computationally too expensive when dealing with high dimensional datasets.

To overcome these limitations, in [5] the authors propose an algorithm that exploits entropic graphs to estimate both the id of a manifold, and the intrinsic entropy of the manifold random samples. This technique is based on the observation that the length function of such graphs, that is the sum of arc weights on the minimal graph that spans all the points in the dataset, is strongly dependent on d. The authors test their method by adopting either the geodesic minimal spanning tree (GMST [4]), where the arc weights are the geodetic distances computed through the ISOMAP [23] algorithm, or the kNN-graph (kNNG [5]), where the arc weights are based on the Euclidean distances, thus requiring a lower computational cost.

The previously described algorithm, exploiting kNNG, can be inserted into the group of approaches, such as LLE [22], NN estimator [20], and TVF [19], that estimate the id by analyzing the relationships among nearest-neighbors within the data. Most of these methods consider hyperspheres with sufficiently small radius r and centered on the points in the dataset, and they estimate some statistics by considering the neighboring points, included into the hypersphere; these statistics are expressed as functions of the intrinsic dimensionality of the manifold from which the points have been randomly drawn. One of these techniques is the Correlation Dimension (CD) estimator [12]; it is based on the assumption that the volume of a d-dimensional set scales with its size r as r^d, which implies that also the number of samples covered by a hypersphere with radius r grows proportionally to r^d. Since the performance of the CD estimator is affected by the choice of the scale r, in [13] the authors suggest an estimator (which we will call Hein) based on the asymptotics of a smoothed version of the CD estimate. Another interesting approach is proposed in [8], where the author presented an algorithm to estimate the id of a manifold in a small neighborhood of a selected point, and they analyzed its finite-sample convergence properties. Another well known technique, based on the analysis of point neighborhoods, is the Maximum Likelihood Estimator (MLE) [17] that applies the principle of maximum likelihood to the distances between close neighbors, and derives the estimator by a Poisson process approximation. More precisely, calling k the number of neighbors, $\boldsymbol{x}_i$ the i-th point, and $T_k(\boldsymbol{x}_i)$ the radius of the smallest sphere centered in $\boldsymbol{x}_i$ containing exactly k neighbors, the local intrinsic dimension is estimated as:

$$\hat{d}(\boldsymbol{x}_i) = \left(\frac{1}{k} \sum_{j=1}^{k} \log \frac{T_{k+1}(\boldsymbol{x}_i)}{T_j(\boldsymbol{x}_i)} \right)^{-1}$$

To our knowledge, most of the neighborhood based estimators generally underestimate d when its value is sufficiently high. To address this problem few techniques have been proposed, among which we recall the method described in [1]. In this work, Camastra et al. propose a correction of the estimated id based on the estimation of the error obtained on synthetically produced datasets of known dimensionality (hypercubes).

3 The Proposed Algorithms

In this section we firstly present our theoretical results (see Section 3.1); according to them, we propose a maximum likelihood `id` estimator and its variants (see Section 3.2); moreover, we present an `id` estimation algorithm based on `pdf` comparison (see Section 3.3) which is more robust than state of the art methods with respect to high dimensional data. For the sake of clarity, in Appendix A the pseudocode of these algorithms is reported.

3.1 Base Theoretical Results

Consider a manifold $\boldsymbol{\mathcal{M}} \equiv \Re^d$ embedded in a higher dimensional space $\Re^D$ through a locally isometric non-linear smooth map $\psi : \Re^d \to \Re^D$; to estimate the `id` of $\boldsymbol{\mathcal{M}}$ we need to identify a "mathematical object", depending only on d, that can be estimated by means of points drawn from the embedded manifold.

To face this problem, we firstly consider the unit hypersphere $\boldsymbol{\mathcal{B}}_d(\mathbf{0}_d, 1) \subset \Re^d$ centered in the origin and uniformly sampled; furthermore, we assume ψ to be the identity map. Considering k points $\{\boldsymbol{z}_i\}_{i=1}^k$ uniformly drawn from $\boldsymbol{\mathcal{B}}_d(\mathbf{0}_d, 1)$, our aim is to find the `pdf` related to the minimum distance between the k points and the hypersphere center $\mathbf{0}_d$.

Call $p(r)$ the `pdf` for the event $\|\boldsymbol{z}_i\| = r$ ($r \in [0, 1]$) where $\|\cdot\|$ is the L_2 norm operator, and denote with $P(\check{r} < r)$ the probability for the event $\|\boldsymbol{z}_i\| < r$; being $\boldsymbol{z}_i$ uniformly drawn it is possible to evaluate these probabilities by means of hypersphere volume ratios. The volume of a d dimensional hypersphere of radius r is:

$$V_r = r^d \frac{\pi^{d/2}}{\Gamma\left(\frac{d}{2}+1\right)} = r^d V_1$$

where $\Gamma(\cdot)$ is the Gamma function and V_1 is the volume of the unit d-dimensional hypersphere. The quantity $P(\check{r} < r)$ is given by the volume ratio $\frac{V_r}{V_1} = r^d$; moreover, being $P(\check{r} < r)$ the cumulative density function (`cdf`) related to the `pdf` $p(r)$, it is $p(r) = \partial \frac{V_r}{V_1} / \partial r = dr^{d-1}$.

The `pdf` $g(r; d, k)$ related to the event $\min_{i \in \{1,\cdots,k\}} \|\boldsymbol{z}_i\| = r$ (i.e. the minimum distance between the points $\{\boldsymbol{z}_i\}_{i=1}^k$ and the hypersphere center equals to r) is proportional to the probability of drawing one point with distance r multiplied by that of drawing $k-1$ points with distance $\check{r} > r$, that is:

$$\begin{aligned} g(r; d, k) \propto \breve{g}(r; d, k) &= p(r)\left(1 - P(\check{r} < r)\right)^{k-1} = \\ &= \frac{\partial \frac{V_r}{V_1}}{\partial r}\left(1 - \frac{V_r}{V_1}\right)^{k-1} = \frac{1}{V_1} d r^{d-1}(1 - r^d)^{k-1} \end{aligned}$$

Normalizing by $\int_0^1 \breve{g}(r; d, k) dr = (V_1 k)^{-1}$ we finally get:

$$g(r; k, d) = \frac{\breve{g}(r; d, k)}{\int_0^1 \breve{g}(r; d, k) dr} = k d r^{d-1}(1 - r^d)^{k-1} \tag{1}$$

Notice that Equation (1) holds only if we assume that the manifold is the unit radius hypersphere. Nevertheless, choosing a d-dimensional open ball $\boldsymbol{\mathcal{B}}_d(\boldsymbol{c}, \epsilon)$ with center $\boldsymbol{c} \in \boldsymbol{\mathcal{M}}$ and radius $\epsilon > 0$, as long as ψ is a non-linear smooth map that preserves distances in $\boldsymbol{\mathcal{B}}_d$ and $\boldsymbol{z}$ is uniformly drawn from $\boldsymbol{\mathcal{B}}_d$, the quantities $\frac{1}{\epsilon}\|\psi(\boldsymbol{c}) - \psi(\boldsymbol{z})\| = \frac{1}{\epsilon}\|\boldsymbol{c} - \boldsymbol{z}\|$ are distributed as the norms of points uniformly drawn from $\boldsymbol{\mathcal{B}}_d(\boldsymbol{0}, 1)$. This fact ensures that Equation (1) holds in $\boldsymbol{\mathcal{B}}_d(\boldsymbol{c}, \epsilon)$ for $r = \frac{1}{\epsilon}\|\boldsymbol{c} - \boldsymbol{z}\|$.

To further generalize our theoretical results, we consider a locally isometric smooth map $\psi : \boldsymbol{\mathcal{M}} \to \Re^D$, and samples drawn from $\boldsymbol{\mathcal{M}} \equiv \Re^d$ by means of a non-uniform smooth `pdf` $f : \boldsymbol{\mathcal{M}} \to \Re^+$. Notice that, being ψ a local isometry, it induces a distance function $\delta_\psi(\cdot, \cdot)$ representing the metric on $\psi(\boldsymbol{\mathcal{M}})$. Under these assumptions Equation (1) does not represent the correct `pdf` of the distances. However, without loss of generality, we consider $\boldsymbol{c} = \boldsymbol{0}_d \in \Re^d$ and $\psi(\boldsymbol{c}) = \boldsymbol{0}_D \in \Re^D$, and we show that any smooth `pdf` f is locally uniform where the probability is not zero. To this aim, assuming $f(\boldsymbol{0}_d) > 0$ and $\boldsymbol{z} \in \Re^d$, we denote with f_ϵ the `pdf` obtained by setting $f_\epsilon(\boldsymbol{z}) = 0$ when $\|\boldsymbol{z}\| > 1$, and $f_\epsilon(\boldsymbol{z}) \propto f(\epsilon \boldsymbol{z})$ when $\|\boldsymbol{z}\| \leq 1$. More precisely, denoting with $\chi_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}$ the indicator function on the ball $\boldsymbol{\mathcal{B}}_d(\boldsymbol{0}, 1)$, we obtain:

$$f_\epsilon(\boldsymbol{z}) = \frac{f(\epsilon \boldsymbol{z}) \chi_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}(\boldsymbol{z})}{\int_{\boldsymbol{t} \in \boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t}} \tag{2}$$

Theorem 1. *Given $\{\epsilon_i\} \to 0^+$, Equation (2) describes a sequence of `pdf` having the unit d-dimensional ball as support; such sequence converges uniformly to the uniform distribution $\mathbf{B}_d$ in the ball $\boldsymbol{\mathcal{B}}_d(\boldsymbol{0}, 1)$.*

Proof. Evaluating the limit for $\epsilon \to 0^+$ of the distance between f_ϵ and $\mathbf{B}_d$ in the supremum norm we get:

$$\begin{aligned}
\lim_{\epsilon \to 0^+} \|f_\epsilon(\boldsymbol{z}) - \mathbf{B}_d(\boldsymbol{z})\|_{\sup} &= \lim_{\epsilon \to 0^+} \left\| \frac{f(\epsilon \boldsymbol{z}) \chi_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}}{\int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t}} - \frac{\chi_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}}{\int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} d\boldsymbol{t}} \right\|_{\sup} \\
\{\textit{just notation}\} &= \lim_{\epsilon \to 0^+} \left\| \frac{f(\epsilon \boldsymbol{z})}{\int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t}} - \frac{1}{\int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} d\boldsymbol{t}} \right\|_{\sup_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}} \\
\left\{ \textit{setting } V = \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} d\boldsymbol{t} \right\} &= \lim_{\epsilon \to 0^+} \left\| \frac{V f(\epsilon \boldsymbol{z}) - \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t}}{V \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t}} \right\|_{\sup_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}} \\
\left\{ 0 < \lim_{\epsilon \to 0^+} V \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t} < \infty \right\} &= \lim_{\epsilon \to 0^+} \left\| V f(\epsilon \boldsymbol{z}) - \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t} \right\|_{\sup_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)}}
\end{aligned}$$

Defining:

$$min(\epsilon) = \min_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{z}) \qquad max(\epsilon) = \max_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{z})$$

and noting that $min(\epsilon) > 0$ definitely since $f(\boldsymbol{0}_d) > 0$, we have:

$$\begin{aligned}
V \cdot min(\epsilon) &\leq \quad V f(\epsilon \boldsymbol{z}) \quad \leq V \cdot max(\epsilon) \\
V \cdot min(\epsilon) &\leq \int_{\boldsymbol{\mathcal{B}}_d(\boldsymbol{0},1)} f(\epsilon \boldsymbol{t}) d\boldsymbol{t} \leq V \cdot max(\epsilon)
\end{aligned}$$

thus their difference is bounded by $V(max(\epsilon) - min(\epsilon)) \xrightarrow[\epsilon \to 0^+]{} 0^+$. □

Theorem 1 proves that the convergence of f_ϵ to $\mathbf{B}_d$ is uniform, and when $\epsilon \to 0^+$ the pdf related to the geodetic distances $\frac{1}{\epsilon}\delta_\psi\left(\psi(\boldsymbol{c}), \psi(\boldsymbol{z})\right) = \frac{1}{\epsilon}\|\boldsymbol{c} - \boldsymbol{z}\|$ converges to the pdf g reported in Equation (1).

3.2 Maximum Likelihood Approaches

Consider a manifold $\boldsymbol{\mathcal{M}} \equiv \Re^d$ embedded in a higher dimensional space $\Re^D$ through a locally isometric non-linear smooth map $\psi : \boldsymbol{\mathcal{M}} \to \Re^D$. Given a sample set $\boldsymbol{X}_N = \{\boldsymbol{x}_i\}_{i=1}^N = \{\psi(\boldsymbol{z}_i)\}_{i=1}^N \subset \Re^D$, where $\boldsymbol{z}_i$ are independent identically distributed points drawn from $\boldsymbol{\mathcal{M}}$ according to a non-uniform smooth pdf $f : \boldsymbol{\mathcal{M}} \to \Re^+$, for each point $\boldsymbol{x}_i \in \boldsymbol{X}_N$ we find the set of $k+1$ ($1 \leq k \leq N-1$) nearest neighbors $\bar{\boldsymbol{X}}_{k+1} = \bar{\boldsymbol{X}}_{k+1}(\boldsymbol{x}_i) = \{\boldsymbol{x}_j\}_{j=1}^{k+1} \subset \boldsymbol{X}_N$. Calling $\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}_{k+1}(\boldsymbol{x}_i) \in \bar{\boldsymbol{X}}_{k+1}$ the most distant point from $\boldsymbol{x}_i$, we calculate the distance between $\boldsymbol{x}_i$ and the nearest neighbor in $\bar{\boldsymbol{X}}_{k+1}$ and we normalize it by means of the distance between $\boldsymbol{x}_i$ and $\hat{\boldsymbol{x}}$. More precisely, we have:

$$\rho(\boldsymbol{x}_i) = \min_{\boldsymbol{x}_j \in \bar{\boldsymbol{X}}_{k+1}} \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}\|} \tag{3}$$

Theorem 4 in [3] ensures that geodetic distances in the infinitesimal ball converge to Euclidean distances with probability 1; moreover, recalling the result reported in Theorem 1, it is possible to notice that, for $\boldsymbol{x}_i \neq \hat{\boldsymbol{x}}$, the quantities $\rho(\boldsymbol{x}_i)$ are samples drawn from the pdf reported in Equation (1), where the parameter k is known and the parameter d must be estimated. A simple approach for the estimation of d is the maximization of the log-likelihood function:

$$\begin{aligned} ll(d) = \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log g(\boldsymbol{x}_i; k, d) = N \log k + N \log d + \\ (d-1) \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log \rho(\boldsymbol{x}_i) + (k-1) \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log\left(1 - \rho^d(\boldsymbol{x}_i)\right) \end{aligned} \tag{4}$$

To select an integer value in $\hat{d} \in \{1..D\}$ as the estimated id, it suffices to evaluate $\hat{d} = \arg\max_{d \in \{1..D\}} ll(d)$; we call this estimator $\texttt{MIND}_{\texttt{MLi}}$. On the other side, if a real value is required as a fractal id estimation, the maximal value in $[1, D]$ must be found. To this aim we compute the first derivative of $ll(d)$ and we determine the solutions of $\frac{\partial ll}{\partial d} = 0$, thus obtaining:

$$\frac{N}{d} + \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \left(\log \rho(\boldsymbol{x}_i) - (k-1) \frac{\rho^d(\boldsymbol{x}_i) \log \rho(\boldsymbol{x}_i)}{1 - \rho^d(\boldsymbol{x}_i)} \right) = 0 \tag{5}$$

We recall that the well-known MLE technique adopts a similar derivation since it extracts distance information from all the first k nearest neighbors. We note that, in the particular case $k = 1$, the solution of Equation (5) is:

$$\hat{d} = -\left(\frac{1}{N} \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log \rho(\boldsymbol{x}_i) \right)^{-1} \tag{6}$$

that is exactly the MLE estimator proposed in [18] when $k = 1$; we call this estimator $\texttt{MIND}_{\texttt{ML1}}$ and its time complexity is $O(DN \log N)$.

For $k > 1$ we numerically solve the following optimization problem:

$$\hat{d} = \arg\max_{1 \leq d \leq D} ll(d) \tag{7}$$

To solve this maximization problem we employed the constrained optimization method proposed in [2] with the initial (integer) value $d_0 = \arg\max_{d \in \{1..D\}} ll(d)$. We call this estimator $\texttt{MIND}_{\texttt{MLk}}$; its time complexity is $O(D^2 N \log N)$.

3.3 A pdf Comparison Approach

In Section 3.2 we presented maximum likelihood estimators for the parameter d (id) in the pdf reported in Equation (1). Notice that, once k is fixed, Equation (1) represents a finite family of D pdfs for all the parameter values $1 \leq d \leq D$. Exploiting this fact, another approach for the estimation of the missing parameter d is the comparison between the D possible theoretical pdfs and a density function estimated by means of the given data.

Consider $\mathcal{M}$ to be a d-dimensional hypersphere embedded in the Euclidean space $\Re^D$; moreover, denote with $\hat{g}(r;k)$ an estimation of $g(r;k,d)$ computed by solely using the sample data points and therefore independent from d. The estimation $\hat{d}$ is computed by choosing the dimensionality which minimizes the Kullback-Leibler divergence between g and $\hat{g}$:

$$\hat{d} = \arg\min_{1 \leq d \leq D} \int_0^1 \hat{g}(r;k) \log \left(\frac{\hat{g}(r;k)}{g(r;k,d)} \right) dr \tag{8}$$

The function $\hat{g}$ can be obtained by means of a set of sample data points as a parametric model; nevertheless, as shown in [6], the number of sample points required to perform dimensionality estimation grows exponentially with the value of the id ("curse of dimensionality"). For this reason, when the dimensionality is too high, the number of sample points practically available is insufficient to compute an acceptable estimation. Moreover, the fraction between the points on (or close to) the edge of the manifold, and the other points (inside the manifold) increases in probability when the dimensionality increases ("edge effect", see [24]), thus affecting the results achieved by estimators based on statistics related to the behavior of point neighborhoods, such as the algorithms proposed in Section 3.2 and MLE.

To address these problems in literature few approaches have been proposed, among which we recall [1]. In this work, Camastra et al. propose a correction of the estimated id based on the estimation of the error obtained on synthetically produced datasets of known dimensionality (hypercubes).

In our work, to reduce the bias between the analytical pdf g and the estimated one $\hat{g}$, for each value $1 \leq d \leq D$ we learn a test pdf $\check{g}_d(r;k)$ by means of points uniformly drawn from the d-dimensional unit hypersphere; moreover, to best resemble the point density of the given dataset, we draw exactly N points per

dimensionality. Finally, we numerically estimate the Kullback-Leibler divergence by means of the estimates $\hat{g}$ and $\check{g}_d$.

More precisely, consider a manifold $\boldsymbol{\mathcal{M}} \equiv \Re^d$ embedded in a higher dimensional space $\Re^D$ through a locally isometric non-linear smooth map $\psi : \boldsymbol{\mathcal{M}} \to \Re^D$. Given a sample set $\boldsymbol{X}_N = \{\boldsymbol{x}_i\}_{i=1}^N = \{\psi(\boldsymbol{z}_i)\}_{i=1}^N \subset \Re^D$ where $\boldsymbol{z}_i$ are independent identically distributed points drawn from $\boldsymbol{\mathcal{M}}$ according to a non-uniform smooth `pdf` $f : \boldsymbol{\mathcal{M}} \to \Re^+$, we compute a vector of normalized distances $\hat{\boldsymbol{r}} = \{\hat{r}_i\}_{i=1}^N = \{\rho(\boldsymbol{x}_i)\}_{i=1}^N$ by means of Equation (3). Moreover, for each dimensionality $d \in \{1..D\}$ we uniformly draw a set of N points $\boldsymbol{Y}_{Nd} = \{\boldsymbol{y}_i\}_{i=1}^N$ from the unit d-dimensional hypersphere, and we similarly compute a vector of normalized distances $\check{\boldsymbol{r}}_d = \{\check{r}_{id}\}_{i=1}^N = \{\rho(\boldsymbol{y}_i)\}_{i=1}^N$. Notice that, a d-dimensional vector randomly sampled from a d dimensional hypersphere according to the uniform `pdf`, can be generated by drawing a point $\bar{\boldsymbol{y}}$ from a standard normal distribution $\mathcal{N}(\cdot|\mathbf{0}_d, 1)$ and by scaling its norm (see Section 3.29 of [9]):

$$\boldsymbol{y} = \frac{u^{\frac{1}{d}}}{\|\bar{\boldsymbol{y}}\|}\bar{\boldsymbol{y}}, \qquad \bar{\boldsymbol{y}} \sim \mathcal{N}(\cdot|\mathbf{0}_d, 1) \tag{9}$$

where u is a random sample drawn from the uniform distribution $U(0, 1)$.

Given a set of values $r_{i=1}^N \subset [0, 1]$ distributed according to the `pdf` p, in [25] the following `pdf` estimator is proposed:

$$\hat{p}(r) = \frac{N^{-1}}{2\rho(r)} \tag{10}$$

where $\rho(r)$ is the distance between r and its nearest neighbor. In our problem, considering a distance $\hat{r}_i \in \hat{\boldsymbol{r}}$, the `pdf` estimates $\hat{g}$ and $\check{g}_d$ can be computed as follows:

$$\hat{g}(\hat{r}_i; k) = \frac{1/(N-1)}{2\hat{\rho}(\hat{r}_i)} \qquad \check{g}_d(\hat{r}_i; k) = \frac{1/N}{2\check{\rho}_d(\hat{r}_i)} \tag{11}$$

where $\hat{\rho}(\hat{r}_i)$ and $\check{\rho}_d(\hat{r}_i)$ are the distances between $\hat{r}_i$ and its first neighbor in $\hat{\boldsymbol{r}}$ and in $\check{\boldsymbol{r}}_d$ respectively.

In [25] a Kullback-Leibler divergence estimator based on the nearest neighbor search is proposed; moreover, the authors show that their method is more effective than partitioning-based techniques, especially when the number of samples is limited. Employing this estimator between $\hat{g}$ and $\check{g}_d$ we obtain:

$$\begin{aligned} \hat{KL}(\hat{g}, \check{g}_d) &= \frac{1}{N}\sum_{i=1}^{N} \log \frac{\hat{g}(\hat{r}_i; k)}{\check{g}_d(\hat{r}_i; k)} = \frac{1}{N}\sum_{i=1}^{N} \log \frac{\frac{1/(N-1)}{2\hat{\rho}(\hat{r}_i)}}{\frac{1/N}{2\check{\rho}_d(\hat{r}_i)}} \\ &= \log \frac{N}{N-1} + \frac{1}{N}\sum_{i=1}^{N} \log \frac{\hat{\rho}(\hat{r}_i)}{\check{\rho}_d(\hat{r}_i)} \end{aligned} \tag{12}$$

Employing Equation (12), the estimated `id` value ($\hat{d}$) is computed as follows:

$$\hat{d} = \underset{d \in \{1..D\}}{\arg\min} \left(\log \frac{N}{N-1} + \frac{1}{N}\sum_{i=1}^{N} \log \frac{\hat{\rho}(\hat{r}_i)}{\check{\rho}_d(\hat{r}_i)} \right) \tag{13}$$

Table 1. Brief description of the 15 synthetic and 3 real datasets, where d is the `id` and D is the embedding space dimension. In the synthetic datasets' name, the number in the subscript refers to the dataset name used by the generator proposed in [13].

Dataset	Name	d	D	Description
Syntethic	$\mathcal{M}_1$	10	11	Uniformly sampled sphere linearly embedded.
	$\mathcal{M}_2$	3	5	Affine space.
	$\mathcal{M}_3$	4	6	Concentrated figure, confusable with a $3d$ one.
	$\mathcal{M}_4$	4	8	Non-linear manifold.
	$\mathcal{M}_5$	2	3	2-d Helix
	$\mathcal{M}_6$	6	36	Non-linear manifold.
	$\mathcal{M}_7$	2	3	Swiss-Roll.
	$\mathcal{M}_8$	12	72	Non-linear manifold.
	$\mathcal{M}_9$	20	20	Affine space.
	$\mathcal{M}_{10a}$	10	11	Uniformly sampled hypercube.
	$\mathcal{M}_{10b}$	17	18	Uniformly sampled hypercube.
	$\mathcal{M}_{10c}$	24	25	Uniformly sampled hypercube.
	$\mathcal{M}_{11}$	2	3	Möebius band 10-times twisted.
	$\mathcal{M}_{12}$	20	20	Isotropic multivariate Gaussian.
	$\mathcal{M}_{13}$	1	13	Curve.
Real	$\mathcal{M}_{\texttt{Faces}}$	3	4096	`ISOMAP` face dataset.
	$\mathcal{M}_{\texttt{MNIST1}}$	$8-11$	784	`MNIST` database (digit 1).
	$\mathcal{M}_{\texttt{SantaFe}}$	9	50	`Santa Fe` dataset (version $D2$).

We call this estimator $\texttt{MIND}_{\texttt{KL}}$; its time complexity is $O(D^2 N \log N)$. To obtain a stable `id` estimation we execute $\texttt{MiND}_{\texttt{KL}}$ 20 times for each dataset and we average the obtained results.

Due to Theorem 1, Theorem 4 in [3], and considering that the Kullback-Leibler divergence estimator employed is consistent as shown in [25], Equation (13) represents a consistent estimator for the intrinsic dimensionality of the manifold $\mathcal{M}$.

4 Algorithm Evaluation

In this section we describe the datasets employed in our experiments (see Section 4.1), we summarize the adopted experimental settings (see Section 4.2), and we report the achieved results (see Section 4.3).

4.1 Dataset Description

To evaluate our algorithms, we have performed experiments on both 15 synthetic and 3 real datasets (see Table 1). To generate the synthetic datasets we have employed the tool proposed in [13]. Instead, the real datasets are the `ISOMAP` face database [23], the `MNIST` database [16] and the `Santa Fe` [21] dataset.

The `ISOMAP` face database consists in 698 gray-level images of size 64×64 depicting the face of a sculpture. This dataset has three degrees of freedom: two for the pose and one for the lighting direction.

The MNIST database consists in 70000 gray-level images of size 28×28 of hand-written digits; in our tests we used the 6742 training points representing the digit 1. The id of this database is not actually known, but some works have proposed similar estimations [13,3] for the different digits; considering digit 1, the proposed id values are in the range $\{8..11\}$.

The version $D2$ of the Santa Fe dataset is a synthetic time series of 50000 one-dimensional points; it was generated by a simulation of particle motion, and it has nine degrees of freedom. In order to estimate the attractor dimension of this time series, we used the method of delays described in [7], which generates D-dimensional vectors by collecting D values from the original dataset; by choosing $D = 50$ we obtained a dataset containing 1000 points in $\Re^{50}$.

4.2 Experimental Setting

We compared our method with well-known id estimators: PCA, kNNG, CD, MLE, and Hein. For kNNG, MLE, and Hein[1] we used the authors' Matlab implementation, whilst for the other algorithms we employed the version provided by the dimensionality reduction toolbox[2].

To generate the synthetic datasets we adopted the generator described in [13] creating 20 instances of each dataset reported in Table 1, each of which is composed by 2000 randomly sampled points. For each technique, to obtain a stable estimation, we averaged the results achieved on the different subsets. To execute multiple tests on $\mathcal{M}_{\texttt{MNIST1}}$, we extracted from the digit 1 dataset 5 random subsets containing 2000 points each.

Table 2. Parameter settings for the different estimators: k represents the number of neighbors, γ the edge weighting factor for kNN, M the number of Least Square (LS) runs, and N the number of resampling trials per LS iterations

Method	Synthetic	Real
PCA	$Threshold = 0.025$	$Threshold = 0.0025$
CD	$None$	$None$
MLE	$k_1 = 6\ k_2 = 20$	$k_1 = 3\ k_2 = 8$
$\texttt{kNNG}_1$	$k_1 = 6, k_2 = 20, \gamma = 1, M = 1, N = 10$	$k_1 = 3, k_2 = 8, \gamma = 1, M = 1, N = 10$
$\texttt{kNNG}_2$	$k_1 = 6, k_2 = 20, \gamma = 1, M = 10, N = 1$	$k_1 = 3, k_2 = 8, \gamma = 1, M = 10, N = 1$
$\texttt{MiND}_{\texttt{ML1}}$	$k = 1$	$k = 1$
$\texttt{MiND}_{\texttt{MLk}}$	$k = 10$	$k = 5$
$\texttt{MiND}_{\texttt{MLi}}$	$k = 10$	$k = 5$
$\texttt{MiND}_{\texttt{KL}}$	$k = 10$	$k = 5$

In Table 2 the employed configuration parameters are summarized. To relax the dependency of kNNG from the selection of its parameter k, we performed

[1] http://www.eecs.umich.edu/∼hero/IntrinsicDim/,
http://www.stat.lsa.umich.edu/∼elevina/mledim.m,
http://www.ml.uni-saarland.de/code.shtml

[2] http://cseweb.ucsd.edu/∼lvdmaaten/dr/download.php

multiple runs with $k_1 \leq k \leq k_2$ (see Table 2) and we averaged the achieved results.

4.3 Experimental Results

In this section the results achieved on both real and synthetic datasets are reported.

In Table 3 the results achieved on the synthetic datasets are summarized. As can be noticed, all the algorithms but `PCA` achieve good results for datasets with low dimensionality ($d < 10$), whilst the `PCA` method obtains bad estimations when dealing with non-linear manifolds, producing overestimates. With high dimensional datasets, all the techniques but `PCA` and $\texttt{MiND}_{\texttt{KL}}$ achieve bad results generally underestimating the `id` due to the edge-effect and the insufficient cardinality of the datasets. Notice that, $\texttt{MiND}_{\texttt{KL}}$ achieves good approximations both on low and high intrinsic dimensional datasets, dealing with both linear and non-linear embeddings, thus obtaining results always comparable with those that better approximate the `id`. In the last row of Table 3 the Mean Percentage Error (`MPE`) indicator is reported; for each algorithm this value is computed as the mean of the percentage errors obtained on each dataset:

$$\texttt{MPE} = \frac{100}{\#\mathcal{M}} \sum_{\mathcal{M}} \frac{|\hat{d}_{\mathcal{M}} - d_{\mathcal{M}}|}{d_{\mathcal{M}}} \tag{14}$$

where $\#\mathcal{M} = 15$ is the number of tested manifolds (see Table 1). Notice that $\texttt{MiND}_{\texttt{KL}}$ obtains the minimum `MPE`.

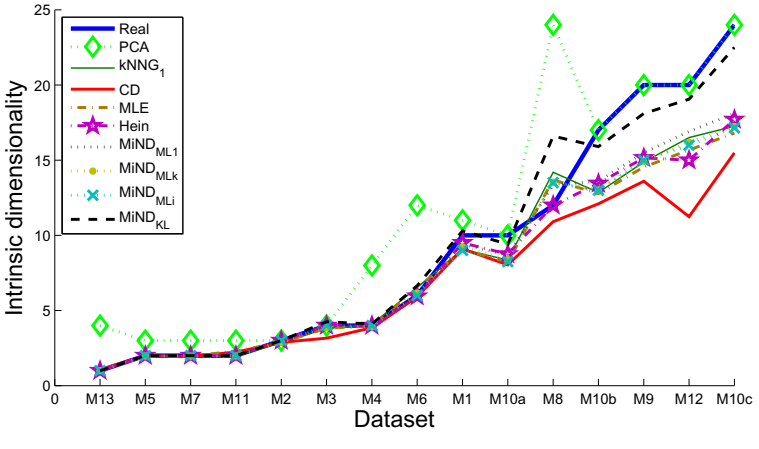


Fig. 1. Comparison between the results achieved by the `id` estimators with respect to the increase of the manifolds' `id`

Table 3. Results achieved on the synthetic datasets. The last row contains the MPE indicator that allows to perform a direct comparison among methods. The best approximations are highlighted in bold case.

Dataset	d	PCA	$kNNG_1$	$kNNG_2$	CD	MLE	Hein	$MiND_{ML1}$	$MiND_{MLk}$	$MiND_{MLi}$	$MiND_{KL}$
$\mathcal{M}_{13}$	*1*	4.00	**1.00**	1.01	1.07	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
$\mathcal{M}_5$	*2*	3.00	1.96	**2.00**	1.98	1.96	**2.00**	1.97	1.97	**2.00**	**2.00**
$\mathcal{M}_7$	*2*	3.00	1.93	1.98	1.94	1.97	**2.00**	1.98	1.96	**2.00**	**2.00**
$\mathcal{M}_{11}$	*2*	3.00	1.96	2.01	2.23	2.30	**2.00**	1.97	1.97	**2.00**	**2.00**
$\mathcal{M}_2$	*3*	**3.00**	2.85	2.93	2.88	2.87	**3.00**	2.93	2.88	**3.00**	**3.00**
$\mathcal{M}_3$	*4*	**4.00**	3.80	4.22	3.16	3.82	**4.00**	3.89	3.84	**4.00**	4.25
$\mathcal{M}_4$	*4*	8.00	4.08	4.06	3.85	3.98	**4.00**	3.95	3.93	**4.00**	4.10
$\mathcal{M}_6$	*6*	12.00	6.53	13.99	5.91	6.45	5.95	5.91	6.17	**6.00**	6.65
$\mathcal{M}_1$	*10*	11.00	9.07	9.39	9.09	9.06	9.50	9.41	9.23	9.00	**10.30**
$\mathcal{M}_{10a}$	*10*	**10.00**	8.35	9.00	8.04	8.22	8.75	8.68	8.38	8.25	9.40
$\mathcal{M}_8$	*12*	24.00	14.19	8.29	10.91	13.69	**12.00**	13.35	13.53	13.50	16.60
$\mathcal{M}_{10b}$	*17*	**17.00**	12.85	15.58	12.09	12.77	13.45	13.59	13.02	13.00	15.90
$\mathcal{M}_9$	*20*	**20.00**	14.87	17.07	13.60	14.54	15.15	15.49	14.90	15.00	18.10
$\mathcal{M}_{12}$	*20*	**20.00**	16.50	14.58	11.24	15.67	15.00	16.91	16.19	16.00	19.05
$\mathcal{M}_{10c}$	*24*	**24.00**	17.26	23.68	15.48	16.80	17.70	18.10	17.24	17.15	22.50
MPE		50.67	11.20	16.23	15.38	12.03	7.65	8.32	10.02	9.14	**6.26**

In Table 4 the results achieved by the tested algorithms is reported[3] with respect to the increase of the manifolds' id. This figure confirms that the PCA estimator cannot correctly deal with non-linear manifolds producing strong estimation errors (overestimates). Moreover, this figure underlines that most of the other techniques are able to produce precise estimations when the id is low, whilst they strongly underestimate when the id is sufficiently high. The only exception is given by the results of $MiND_{KL}$; indeed, this method allows to obtain stable estimations both on low and high id datasets.

Table 4. Results achieved on the real datasets by the employed approaches. The best approximations are highlighted in bold case.

Dataset	d	PCA	$kNNG_1$	$kNNG_2$	CD	MLE	Hein	$MiND_{ML1}$	$MiND_{MLk}$	$MiND_{MLi}$	$MiND_{KL}$
$\mathcal{M}_{\text{Faces}}$	*3*	21.00	3.60	4.32	3.37	4.05	**3.00**	3.52	3.59	4.00	3.90
$\mathcal{M}_{\text{MNIST1}}$	*8-11*	11.80	10.37	9.58	6.96	10.29	8.00	11.33	10.02	**9.45**	11.00
$\mathcal{M}_{\text{Santa Fe}}$	*9*	18.00	7.28	7.43	4.39	7.16	6.00	6.31	6.78	7.00	**7.60**

In Table 4 the results achieved on real datasets have been summarized. Notice that also in the case of noisy real datasets, our methods have obtained either the best approximation of the id, or results always comparable with those achieved by the best performing technique.

[3] The results of $kNNG_2$ are omitted for clarity; notice that, this technique has obtained worst results with respect to $kNNG_1$.

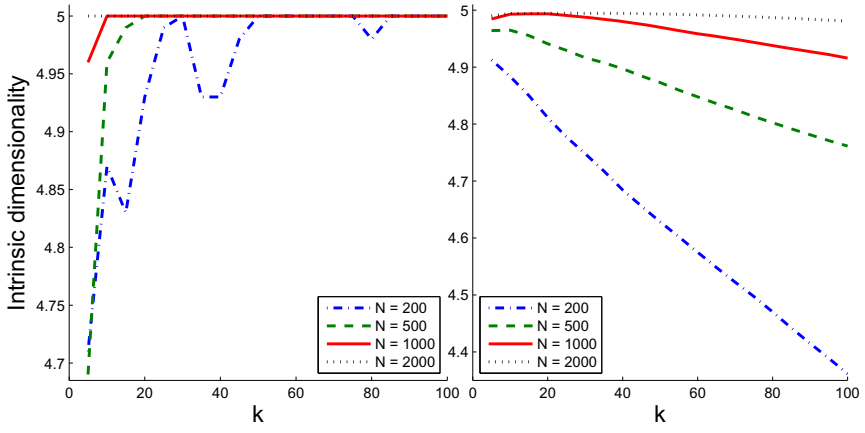


Fig. 2. Behavior of $\texttt{MiND}_{\texttt{KL}}$ (left) and $\texttt{MiND}_{\texttt{MLk}}$ (right) applied to points drawn from a 5-dimensional standard normal distribution, in this test $N \in \{200, 500, 1000, 2000\}$ and $k \in \{5..100\}$

Finally, to test the robustness of our algorithms with respect to the choice of the parameter k, we reproduced the experiments proposed for the `MLE` algorithm in Figure 1 (a) of [17] employing $\texttt{MiND}_{\texttt{KL}}$ and $\texttt{MiND}_{\texttt{MLk}}$, and we averaged the curves obtained by 10 runs. In these tests the adopted datasets are composed by points drawn from the standard Gaussian `pdf` in $\Re^5$. We repeated the test for datasets with cardinalities $N \in \{200, 500, 1000, 2000\}$, and varying the parameter k in the range $\{5..100\}$. As shown in 2, $\texttt{MiND}_{\texttt{KL}}$ (left) demonstrates to be robust to the choice of its parameter k, whilst $\texttt{MiND}_{\texttt{MLk}}$ (right) shows a behavior comparable to that of `MLE` (see Figure 1 (a) in [17]).

Concluding, the results achieved on both real and synthetic datasets have confirmed the quality of the proposed methods; more specifically, $\texttt{MiND}_{\texttt{KL}}$ has proved to be the best estimator since it is robust to the choice of its parameter, it obtains the smallest `MPE` (see Table 3), and it achieves good approximations both on high and low `id` datasets.

5 Conclusions and Future Works

In this work we proposed Minimum Neighbor Distance estimators of intrinsic dimension. These methods, for each point in the dataset, exploit the `pdf` related to the normalized distance of its nearest neighbor. More precisely, the algorithms $\texttt{MiND}_{\texttt{ML*}}$ are based on the maximization of the log-likelihood function associated to the normalized `pdf` of the distances, whilst the estimator $\texttt{MiND}_{\texttt{KL}}$ compares the estimated `pdf` with those of random points uniformly drawn from unitary hyperspheres with dimensionality in $\{1..D\}$.

We tested our algorithms on synthetic and real datasets comparing them with state of the art `id` estimators. The achieved results demonstrate that our

techniques are promising, and the Mean Percentage Error indicator underlines the stability of the proposed methods. We note that, among the techniques we described, our experiments demonstrate that $\texttt{MiND}_{\texttt{KL}}$ is the most robust estimator, since it deals with both low and high `id`, and it manages both linear and non-linear manifolds, achieving results always comparable to the best estimations. Furthermore, its performances are not strongly affected by the choice of its only parameter k, and it shows to be resistant to the curse of dimensionality.

In Equation (3) the `kNN` Euclidean distances are normalized by means of the maximum one. In the limit this normalization factor converges to the geodetic one, but for a finite set of sample points an approximation is introduced. In future works we want to investigate the behavior of this approximation to reduce its negative effect on the `id` estimation.

References

1. Camastra, F., Vinciarelli, A.: Estimating the intrinsic dimension of data with a fractal-based method. IEEE Trans. Pattern Anal. Mach. Intell. 24, 1404–1407 (2002)
2. Coleman, T.F., Li, Y.: An interior, trust region approach for nonlinear minimization subject to bounds. SIAM Journal on Optimization 6, 418–445 (1996)
3. Costa, J., Hero, A.O.: Learning intrinsic dimension and entropy of shapes. In: Krim, H., Yezzi, T. (eds.) Statistics and analysis of shapes. Birkhäuser, Basel (2005)
4. Costa, J.A., Hero, A.O.: Geodesic entropic graphs for dimension and entropy estimation in manifold learning. IEEE Transaction on Signal Processing 52(8), 2210–2221 (2004)
5. Costa, J.A., Hero, A.O.: Learning intrinsic dimension and entropy of high-dimensional shape spaces. In: EUSIPCO (2004)
6. Eckmann, J.-P., Ruelle, D.: Fundamental limitations for estimating dimensions and lyapunov exponents in dynamical systems. Physica D: Nonlinear Phenomena 56(2-3), 185–187 (1992)
7. Edward, O.: Chaos in Dynamical Systems. Cambridge University Press, Cambridge (1993)
8. Farahmand, A.M., Szepesvari, C., Audibert, J.Y.: Manifold-adaptive dimension estimation. In: Proc. of ICML (2007)
9. Fishman, G.S.: Monte Carlo: Concepts, Algorithms, and Applications. Springer Series in Operations Research. Springer, New York (1996)
10. Fukunaga, K.: An algorithm for finding intrinsic dimensionality of data. IEEE Trans. on Computers 20, 176–183 (1971)
11. Fukunaga, K.: Intrinsic Dimensionality Extraction. In: Krishnaiah, P.R., Kanal, L.N. (eds.) Classification, Pattern Recognition and Reduction of Dimensionality. North-Holland, Amsterdam (1982)
12. Grassberger, P., Procaccia, I.: Measuring the strangeness of strange attractors. Physica D: Nonlinear Phenomena 9, 189–208 (1983)
13. Hein, M.: Intrinsic dimensionality estimation of submanifolds in euclidean space. In: ICML, pp. 289–296 (2005)
14. Jollife, I.T.: Principal Component Analysis. Springer Series in Statistics. Springer, New York (1986)
15. Kégl, B.: Intrinsic dimension estimation using packing numbers. In: Becker, S., Thrun, S., Obermayer, K. (eds.) NIPS, pp. 681–688. MIT Press, Cambridge (2002)

16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
17. Levina, E., Bickel, P.J.: Maximum likelihood estimation of intrinsic dimension. Ann. Arbor. MI 1, 777–784 (2005)
18. MacKay, D.J.C., Ghahramani, Z.: Comments on 'maximum likelihood estimation of intrinsic dimension' by E. Levina and P. Bickel (2005), http://www.inference.phy.cam.ac.uk/mackay/dimension/
19. Mordohai, P., Medioni, G.: Dimensionality estimation, manifold learning and function approximation using tensor voting. J. Mach. Learn. Res. 11, 411–450 (2010)
20. Pettis, K., Bailey, T., Jain, A., Dubes, R.: An intrinsic dimensionality estimator from near-neighbor information (1979)
21. Pineda, F.J., Sommerer, J.C.: Estimating generalized dimensions and choosing time delays: A fast algorithm. In: Forecasting the Future and Understanding the Past. Time Series Prediction, pp. 367–385 (1994)
22. Roweis, S.T., Saul, L.K.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science 290, 2323–2326 (2000)
23. Tenenbaum, J.B., Silva, V., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science 290, 2319–2323 (2000)
24. Verveer, P.J., Duin, R.P.W.: An evaluation of intrinsic dimensionality estimators. IEEE Transactions on Pattern Analysis and Machine Intelligence 17, 81–86 (1995)
25. Wang, Q., Kulkarni, S.R., Verd, S.: A nearest-neighbor approach to estimating divergence between continuous random vector. In: IEEE Int. Symp. Information Theory (ISIT 2006), pp. 242–246 (2006)

Appendix A **Algorithms Implementation**

In this appendix the pseudocode of our algorithms is reported. In Algorithm 1 $\texttt{MiND}_{\texttt{MLi}}$ is shown, where $kNN(\boldsymbol{X}_N, \boldsymbol{x}, k)$ is the procedure that employs a k-nearest neighbor search returning the set of the k nearest neighbors of $\boldsymbol{x}$ in $\boldsymbol{X}_N$. Since $\texttt{MiND}_{\texttt{ML1}}$ consists only in Equation (6), whilst $\texttt{MiND}_{\texttt{MLk}}$ consists in refining the $\texttt{MiND}_{\texttt{MLi}}$ result by means of a constrained optimization algorithm (see Equation (7)), and due to the lack of space, these two algorithms are not reported. Algorithm 2 reports the pseudocode of $\texttt{MiND}_{\texttt{KL}}$, where $NN(\boldsymbol{X}_N, \boldsymbol{x})$ is the procedure that returns the nearest neighbor of $\boldsymbol{x}$ in $\boldsymbol{X}_N$.

Algorithm 1. Pseudocode for the $\texttt{MiND}_{\texttt{MLi}}$ algorithm.

Input:
  $\boldsymbol{X}_N$: The dataset points $\{\boldsymbol{x}_i\}_{i=1}^N$.
  k: The kNN parameter.
Output:
  $\hat{d}$: The estimated intrinsic dimensionality.
{Compute for each point the normalized radii}
for i:=1 **to** N **do begin**
  $\bar{\boldsymbol{X}}_{k+1} = kNN(\boldsymbol{X}_N, \boldsymbol{x}_i, k)$; {Finding the k neighbors of $\boldsymbol{x}_i$.}
  $\rho(\boldsymbol{x}_i) = \min_{\boldsymbol{x}_j \in \bar{\boldsymbol{X}}_{k+1}} \|\boldsymbol{x}_i - \boldsymbol{x}_j\| / \max_{\hat{\boldsymbol{x}} \in \bar{\boldsymbol{X}}_{k+1}} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}\|$;
end
{Choosing $\hat{d} \in \{1..D\}$ that maximizes the log likelihood}
$\hat{d} = \arg\max_{d \in \{1..D\}} \left((d-1) \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log \rho(\boldsymbol{x}_i) + (k-1) \sum_{\boldsymbol{x}_i \in \boldsymbol{X}_N} \log \left(1 - \rho^d(\boldsymbol{x}_i)\right) \right)$;

Algorithm 2. Pseudocode for the $\texttt{MiND}_{\texttt{KL}}$ algorithm.

Input:
  $\boldsymbol{X}_N$: The dataset points $\{\boldsymbol{x}_i\}_{i=1}^N$.
  k: The kNN parameter.
Output:
  $\hat{d}$: The estimated intrinsic dimensionality.
{Compute for each point the normalized radii}
for i:=1 **to** N **do begin**
  $\bar{\boldsymbol{X}}_{k+1} = kNN(\boldsymbol{X}_N, \boldsymbol{x}_i, k)$; {Finding the k neighbors of $\boldsymbol{x}_i$ in $\boldsymbol{X}_N$.}
  $\hat{r}_i = \rho(\boldsymbol{x}_i) = \min_{\boldsymbol{x}_j \in \bar{\boldsymbol{X}}_{k+1}} \|\boldsymbol{x}_i - \boldsymbol{x}_j\| / \max_{\hat{\boldsymbol{x}} \in \bar{\boldsymbol{X}}_{k+1}} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}\|$;
  {Computing the distance between $\hat{r}_i$ and the NN}
  $\hat{\rho}(\hat{r}_i) = |\hat{r}_i - NN(\{\hat{r}_j\}_{j \neq i}, \hat{r}_i)|$;
end
{Estimate the Kullback Leibler divergences}
for d:=1 **to** D **do begin**
  {Uniformly sampling from the unit ball}
  $\boldsymbol{Y}_{Nd} = \{\boldsymbol{y}_i = \bar{\boldsymbol{y}} u^{1/d} / \|\bar{\boldsymbol{y}}\|; \bar{\boldsymbol{y}} \sim \mathcal{N}(\cdot | \boldsymbol{0}_d, 1), u \sim U(0,1)\}_{i=1}^N$;
  {Compute for each point the normalized radii}
  for i:=1 **to** N **do begin**
    $\bar{\boldsymbol{Y}}_{k+1} = kNN(\boldsymbol{Y}_{Nd}, \boldsymbol{y}_i, k)$;
    $\check{r}_i = \rho(\boldsymbol{y}_i) = \min_{\boldsymbol{y}_j \in \bar{\boldsymbol{Y}}_{k+1}} \|\boldsymbol{y}_i - \boldsymbol{y}_j\| / \max_{\hat{\boldsymbol{y}} \in \bar{\boldsymbol{Y}}_{k+1}} \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}\|$;
  end
  {Computing the distances $\check{\rho}_d(\hat{r}_i)$}
  for i:=1 **to** N **do begin**
    {Computing the distance between $\check{r}_i$ and the NN}
    $\check{\rho}_d(\hat{r}_i) = |\check{r}_i - NN(\{\check{r}_j\}_{j=1}^N, \hat{r}_i)|$;
  end
end
{Estimating the intrinsic dimensionality}
$\hat{d} = \arg\min_{d \in \{1..D\}} \left(\log \frac{N}{N-1} + \frac{1}{N} \sum_{i=1}^N \log \frac{\hat{\rho}(\hat{r}_i)}{\check{\rho}_d(\hat{r}_i)} \right)$

Graph Evolution via Social Diffusion Processes

Dijun Luo, Chris Ding, and Heng Huang

Department of Computer Science and Engineering,
University of Texas, Arlington, Texas, USA
dijun.luo@gmail.com, {chqding,heng}@uta.edu

Abstract. We present a new stochastic process, called as Social Diffusion Process (SDP), to address the graph modeling. Based on this model, we derive a graph evolution algorithm and a series of graph-based approaches to solve machine learning problems, including clustering and semi-supervised learning. SDP can be viewed as a special case of *Matthew effect*, which is a general phenomenon in nature and societies. We use social event as a metaphor of the intrinsic stochastic process for broad range of data. We evaluate our approaches in a large number of frequently used datasets and compare our approaches to other state-of-the-art techniques. Results show that our algorithm outperforms the existing methods in most cases. We also applying our algorithm into the functionality analysis of microRNA and discover biologically interesting cliques. Due to the broad availability of graph-based data, our new model and algorithm potentially have applications in wide range.

1 Introduction

Data clustering, assignment, and dimensional reduction have been the focuses for exploring unknown data [1,2]. Among them, graph-based data analysis techniques have recently been investigated extensively in traditional machine learning problems. One reason for the popularity of graph-based approaches is the broad availability of graph data. For example, social objects (users, blog items, photos) are generated with relational links, and for objects represented in Euclidean space, one can easily obtain a graph by using similarity measurements (*e.g.* Gaussian kernels). Graph-based approaches fall into two categories. The first one is *spectral graph partitioning* methods which address the group detection problem by identifying an approximately minimal set of edges to remove from the graph to achieve a given number of groups [3,4,5,6]. Impressive results have been shown in these methods which have been applied in many practical applications. These approaches relax NP-hard combinatorial problems into continuous optimization problems which can be solved by eigenvector decompositions.

Another approach category is *stochastic modeling*. In stochastic models, the observed data are assumed to be drawn from some distribution and generative assumptions [7,8,9,10,11]. These approaches often lead to a maximum likelihood problems that can be solved by Expectation Maximization (EM) or approximately Variational EM algorithms [12].

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 390–404, 2011.

Among these models, the Chinese Restaurant Processes (CRPs) consider a sequence of customers coming to a restaurant according to the convention of Chinese people: one tends to stay in a place where there are more people. Each customer seeks some previously occupied table and the probability is proportional to the number of customers already sitting there. The new customer also sits in a new table with probability proportional to some parameter. CRP and its variations have been theoretically and empirically studied in many previous researches [10,11,13,14]

In a CRP mixture, customers are data points, and customers sitting at the same table belong to the same cluster. Since the number of occupied tables is random, the resulting posterior distribution of seating assignments provides a distribution of clusterings where the number of clusters is determined by the data.

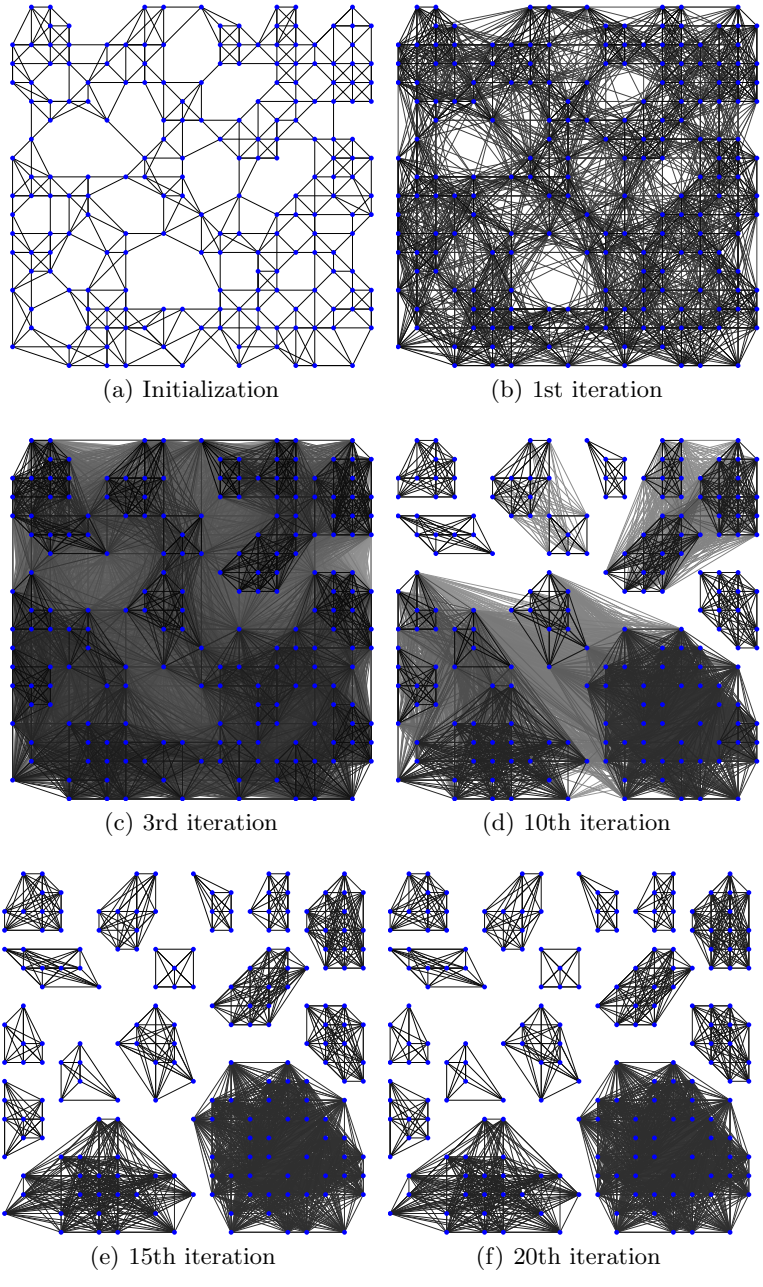
In this paper, we propose a novel stochastic process which further considers the social events among social members as a metaphor of the intrinsic stochastic process for broad range of data. We call this process as Social Diffusion Process. The basic assumption in this model is that two social members tend to communicate if they are familiar with each other or have many common friends, and that the more times they communicate, the more they are familiar with each other.

Based on our model, we derive an iterative evolution algorithm to model the social structures of the members. The major characteristic of our algorithm which differs from most of previous research is that we do not need to impose latent variables which leads to maximum likelihood estimation. Instead, our evolutionary algorithm iteratively generates a new relational graph among social members in which the social structures become more and more clear, please see Figure 1 for a toy example. In this example, our algorithm starts from a random binary network and ends with clearly separated subgraphs. Details can be found in Section 3.2.

The similar algorithm which is closest to our intuition is Markov Clustering (MCL) [15] from the point of view of graph evolution. However, MCL is not suitable for the purpose in this paper. We perform the MCL evolution on the same graph in Figure 1 (a) and the results for MCL are demonstrated in Figure 2. One can observe that the result in Figure 1 is much more reasonable than that in Figure 2.

The results of the evolution algorithm can be viewed as a special case of the *Matthew effect*, in which "The rich get richer". This is a general phenomenon in nature and societies [16,17,18]. One interesting observation in our algorithm is that the evolution of a graph by the SDP enhance the *qualities* of the graph in a wide range of applications. This phenomenon suggests that the SDP assumptions are natural in general. Due to the broad availability of graph-based data, our new model and algorithm have potential applications in various areas.

In the rest of the paper, we first introduce the Social Diffusion Process in Section 2 and the derived algorithm in Section 3. In section 4, we show the evidence of improvement of the quality by our algorithm using extensive experiments.

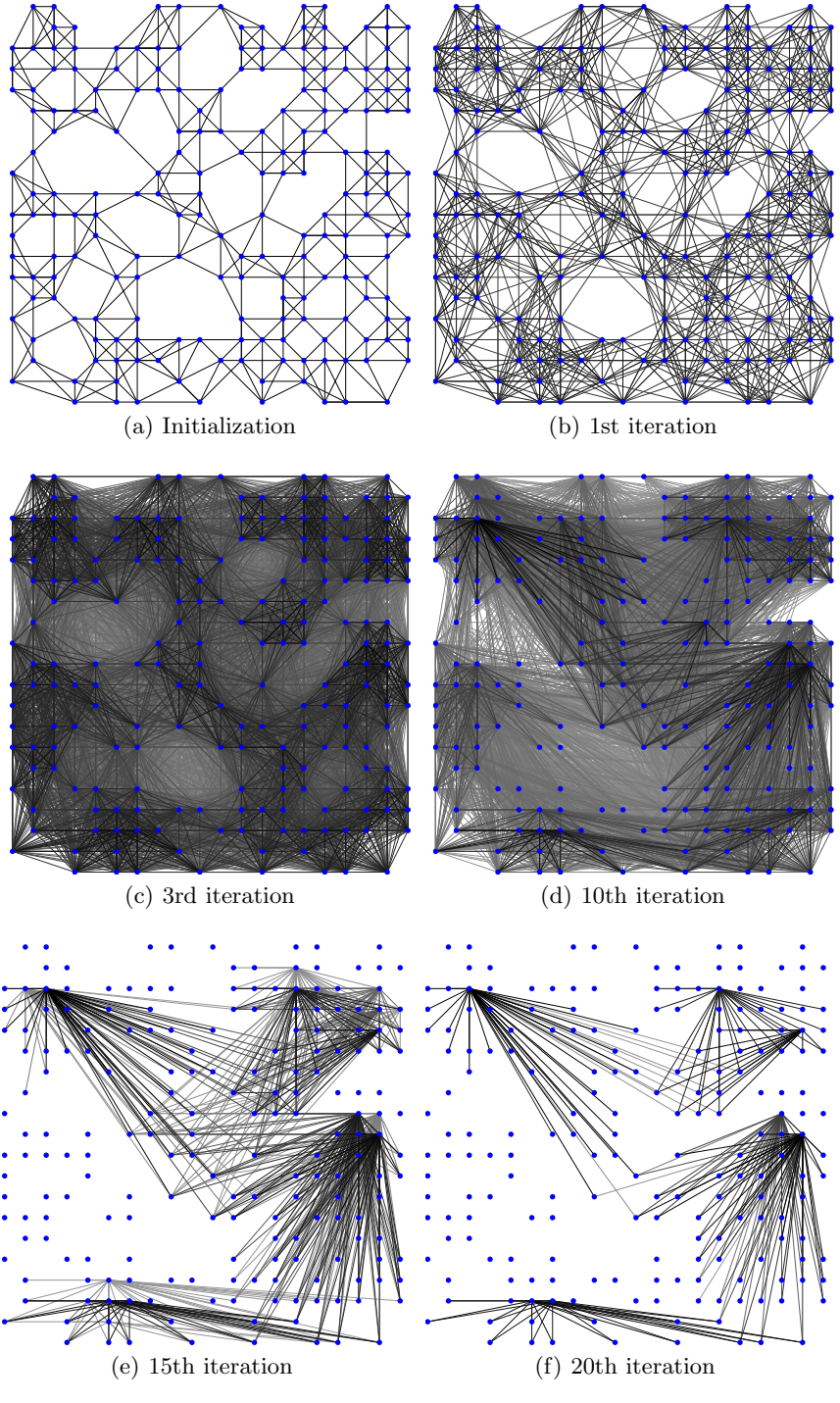


(a) Initialization (b) 1st iteration

(c) 3rd iteration (d) 10th iteration

(e) 15th iteration (f) 20th iteration

Fig. 1. Graph evolution results on the grid toy data based on Social Diffusion Process. Each point (blue dot) represents a social member and the edge between two social members represents the familiarness between them. (a): the original graph. (b)– (f): the condensation results of the 1st, 3rd, 10th, 15th, and 20th iterations of our evolution algorithm. The darkness of the edge represents the familiarness between the social members (the darker the higher).



(a) Initialization (b) 1st iteration

(c) 3rd iteration (d) 10th iteration

(e) 15th iteration (f) 20th iteration

Fig. 2. Graph evolution results on the grid toy data based on Markov Clustering

2 Social Diffusion Process for Friendship Broadening

In this section we introduce the Social Diffusion Process based on the notations of graph.

2.1 Preliminaries

Let $G = \{V, W\}$ denote an undirected weighted graph, where $V = \{v_1, v_2, \cdots, v_n\}$ is the set of nodes, $W \in \mathbb{R}^{n \times n}$ is a $n \times n$ matrix, and W_{ij} denotes the weight of the edge between nodes v_i and v_j. $W_{ij} = 0$, if there is no edge between v_i and v_j.

2.2 Social Events and Broadening of Friendship

We consider the following scenario: A and B are friends. Suppose A brings a friend A_f and meets with B. Now A_f and B become known to each other. If B also brings a friend B_f to the meeting, i.e., the four (A, A_f, B, B_f) meet. Then A_f become known to both B also B_f, i.e., the friendship circle for A_f is broadened. This happens to A, B, B_f as well.

In graph terminology, the initial friendship between A and B is represented by an edge connecting A and B. The broadened friendship between A_f and B (assuming they are not connected at initial stage) has a connection strength somewhere between 0 and 1. In other words, if two persons C and D don't know each other, the existence of a mutual friend connects C and D. Further more, even if A and B are friends (i.e., an edge exists between A and B), their friendship is further enhanced due to the existence of mutual friends. Our main goal is to formally define this friendship broadening process and compute the **friendship enhancement probability**. We expect this enhanced friendship provide a more clear social community structure as shown in Figure 1.

Formally, we define the following events among social members: (1) **Date** (v_i, v_j): v_i and v_j initial a dating. (2) **Bring**(v_i, v_k): v_i brings v_k after the event **Date**(v_i, v_j) for some j. (3) **Meet**(v_p, v_q): v_p and v_q meet in the same table.

We further impose the following rules: (1) If **Date**(v_i, v_j) happens, **Meet**(v_i, v_j) happens, or (2) If **Date**(v_i, v_j) and **Bring**(v_i, v_k) happen, **Meet**(v_k, v_j) happens. (3)If **Date**(v_i, v_j), **Bring**(v_i, v_k), and **Bring**(v_j, v_l) happen, **Meet**(v_j, v_l) happens.

Here we assume **Date**(v_i, v_j) is equivalent to **Date**(v_j, v_i) and **Meet**(v_k, v_l) is equivalent to **Meet**(v_l, v_k).

We use the following to denote the rules above

$$\text{Rule 1:} \quad \mathbf{Date}(v_i, v_j) \Rightarrow \mathbf{Meet}(v_i, v_j) \tag{1}$$

$$\text{Rule 2:} \quad \left.\begin{array}{l} \mathbf{Date}(v_i, v_j) \\ \mathbf{Bring}(v_i, v_k) \end{array}\right\} \Rightarrow \mathbf{Meet}(v_j, v_k) \tag{2}$$

$$\text{Rule 3:} \quad \left.\begin{array}{l} \mathbf{Date}(v_i, v_j) \\ \mathbf{Bring}(v_i, v_k) \\ \mathbf{Bring}(v_j, v_l) \end{array}\right\} \Rightarrow \mathbf{Meet}(v_k, v_l) \tag{3}$$

2.3 Social Diffusion Process

Now we are ready to introduce the Social Diffusion Process. The process starts with a graph $G = \{V, W\}$ where $V = \{v_1, v_2, \cdots, v_n\}$ denotes a set of social members and W denotes the familiarness between social members, *i.e.* W_{ij} represents the familiarness between v_i and v_j, $i, j = 1, 2, \cdots, n$. We assume that $W_{ij} = W_{ji}$. The SDP happens as following,

(1) Choose a threshold $t \sim U(0, \mu)$ where $\mu = \max_{ij} W_{ij}$ and U denotes the uniform distribution.
(2) **Date**(v_i, v_j) happens with a constant probability δ if $W_{ij} \geq t$.
(3) **Bring**(v_i, v_k) and **Bring**(v_j, v_l) happen with probability $p(i, k, t)$, $p(j, l, t)$, respectively, where

$$p(i,k,t) = \begin{cases} \frac{1}{|\mathcal{N}_{i,t}|} & \text{if } v_k \in \mathcal{N}_{i,t} \\ 0 & \text{otherwise} \end{cases},$$

$$p(j,l,t) = \begin{cases} \frac{1}{|\mathcal{N}_{j,t}|} & \text{if } v_k \in \mathcal{N}_{j,t} \\ 0 & \text{otherwise} \end{cases},$$

$\mathcal{N}_{i,t} = \{q : W_{iq} \geq t\}, \mathcal{N}_{j,t} = \{q : W_{jq} \geq t\}$, and $|\cdot|$ denotes the cardinality of the set.
(4) Apply rules (1)–(3). For any p, q, if **Meet**(v_p, v_q), $W_{pq} \leftarrow W_{pq} + \alpha\mu$.

The threshold t can be interpreted as the importance of the dating event. Two friends do not date if they are not familiar with each other enough (thresholded by t)[1]. When a social member brings some friend, he/she only considers those friends who are familiar enough with (thresholded by t). The set $\mathcal{N}_{i,t}$ is the friends the social member v_i can bring with this threshold t. Eq. (4) indicates that social member v_i chooses friends in $\mathcal{N}_{i,t}$ with uniform distribution. Notice that there are two parameters in this model δ and α. In section 3, we will introduce an algorithm based on the SDP, in which the two parameters can be eliminated by natural normalization.

3 Graph Evolution Based on Social Diffusion Process

3.1 The Evolution Algorithm

We first denote A^t as the following

$$(A^t)_{ij} = \begin{cases} 1 & \text{if } W_{ij} \geq t \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

[1] The reason why we use a thresholding of W_{ij} instead of directly using W_{ij} for event **Date**(v_i, v_j) is following. Assume we want to date with some one on the wedding of Royal wedding for William and Kate, who are we going to date? Probably one of our most important friends. In the same event, if we want to bring guest to meet our friend in the date, who are we going to bring? Probably another one of our most important friends. In reality, social events happen according to their importance, denoted as threshold t in the paper. We believe this model is much accurate than directly using W_{ij} as the probability of **Date**(v_i, v_j).

where t is a positive threshold. Consider two social members v_i and v_j. The events in which they meet each other can be divided into three cases:

Case (1). $\mathbf{Date}(v_i, v_j)$. In this case the probability that they meet is

$$P(\mathbf{Meet}(v_i, v_j)) = \delta(A^t)_{ij}.$$

Case (2). $\mathbf{Date}(v_i, v_k)$ and $\mathbf{Bring}(v_k, v_j)$. By definition $|\mathcal{N}_{k,t}| = \sum_j A^t_{jk} = d^t_k$, where d^t_k is the degree k in A^t. In this case,

$$\begin{aligned} & P(\mathbf{Meet}(v_i, v_j)) \\ =& \sum_k P(\mathbf{Meet}(v_i, v_j) | \mathbf{Date}(v_i, v_k), \mathbf{Bring}(v_k, v_j)) \\ =& \sum_k \delta(A^t)_{ik} \frac{A^t_{jk}}{d_k} = \delta(A^t D^{-1} A^t)_{ij}, \end{aligned}$$

where $D = \mathbf{diag}(d_1, d_2, \cdots, d_n)$.

Case(3). $\mathbf{Date}(v_k, v_l)$, $\mathbf{Bring}(v_k, v_i)$, and $\mathbf{Bring}(v_l, v_j)$. Similar with case (2), we have

$$\begin{aligned} P(\mathbf{Meet}(v_i, v_j)) &= \sum_{kl} \delta(A^t)_{kl} \frac{A^t_{ik}}{d_k} \frac{A^t_{jl}}{d_l} \\ &= \delta(A^t D^{-1} A^t D^{-1} A^t)_{ij}. \end{aligned}$$

By summing up the three cases, we have

$$\begin{aligned} & P(\mathbf{Meet}(v_i, v_j)) \\ =& \delta A^t_{ij} + \delta(A^t D^{-1} A^t)_{ij} + \delta(A^t D^{-1} A^t D^{-1} A^t)_{ij}. \end{aligned}$$

From the definition of updating of W, we have

$$\begin{aligned} & \mathbb{E}(\Delta W_{ij}) \\ =& \alpha\mu\delta \left(A^t_{ij} + (A^t D^{-1} A^t)_{ij} + (A^t D^{-1} A^t D^{-1} A^t)_{ij} \right) \\ \triangleq& \alpha\mu\delta M^t_{ij}. \end{aligned} \tag{5}$$

Here $A^t_{ij} + (A^t D^{-1} A^t)_{ij} + (A^t D^{-1} A^t D^{-1} A^t)_{ij}$ is denoted by M^t_{ij}. This suggests that the expectation $\mathbb{E}(\Delta W_{ij})$ is proportional to M^t_{ij}. In our implementation we normalize M^t_{ij} by $M^t_{ij} \leftarrow M^t_{ij} / \sum_{i'j'} M^t_{i'j'}$, which leads to the following algorithm,

In this algorithm, we use an evenly distributed threshold t to approximate the uniform distribution from which t should be drawn from. In our experiments, we set $T = 50$. One should notice that no matter what the choice of the normalization is, the algorithm has the following properties.

Algorithm 1. $\tilde{W} = \textbf{GraphEvolution}(W)$

Input: Graph W
Output: Graph $\tilde{W}$
$\mu = \max_{ij} W_{ij}, \tilde{W} = \mathbf{0}$
for $i = 1 : T$ **do**
  $t = i\mu/T$
  Calculate M^t using Eq. (5)
  Normalize $M^t : M^t_{ij} \leftarrow M^t_{ij} / \sum_{i'j'} M^t_{i'j'}$
  $\tilde{W} \leftarrow \tilde{W} + M^t$
end for
Output: $\tilde{W}$

Property 1. The result of **GraphEvolution** is scale invariant, *i.e.* $\forall \beta > 0$,

$$\textbf{GraphEvolution}(W) = \textbf{GraphEvolution}(\beta W).$$

This is because the threshold t is always evenly distributed in the interval $[0, \max_{ij} W_{ij}]$ and M^t remains the same. In other words, the choice of the normalization does not change any terms in M^t.

Property 2. If W is a set of disconnected full cliques with same size and same weight, *i.e.* there is a partition $\Pi = \{\pi_1, \pi_2, \cdots, \pi_K\}, \pi_k \cap \pi_l = \Phi, 1 \leq k, l \leq K, \cup_k \pi_k = \{v_1, v_2, \cdots, v_n\}$ such that $\forall i, j \in \pi_k, W_{ij} = c$ where c is a constant, and $\forall i \in \pi_k, j \in \pi_l, k \neq l, W_{ij} = 0$, then

$$W \propto \textbf{GraphEvolution}(W).$$

This is easy to show since if W is a set of disconnected full cliques with the same weight, A^t is the same for every $t : A^t_{ij} = 1$ if $A_{ij} \neq 0$, $A^t_{ij} = 0$ otherwise. Thus $M^t \propto W$, which leads to $W \propto \textbf{GraphEvolution}(W)$. This property shows a hint of conditions in which the algorithm of $W \leftarrow \textbf{GraphEvolution}(W)$ converges, which will be discussed later.

3.2 Application of Graph Evolution

The algorithm **GraphEvolution** can be used in different purposes. The basic idea is that it improves the quality in terms of the natural structure underlying the graph data. In this paper, we investigate two applications: clustering and semi-supervised learning.

For the purpose of clustering, one can simply iteratively perform the following

$$W \leftarrow \textbf{GraphEvolution}(W). \tag{6}$$

As iterations continue, the structures of the graph is clearer and clearer. We show results of the evolution algorithm on a toy grid data, see Figure 1.

In this example, we randomly generate 198 points in a 20×20 grid. We obtain an unweighted graph as follows. If node i is one of K-nearest neighbors of node j, or node j is one of the K-nearest neighbors of node i, we set $W_{ij} = 1$, and $W_{ij} = 0$ otherwise. $K = 7$ in this example and the neighborhood is computed using the Euclidean distance of the nodes on the 2-dimensional grid coordinate. The original graph is shown in Figure 2(a).

Starting from this graph, we run the **GraphEvolution** algorithm for 20 iterations and the results of the first, third, 10th, 15th, and 20th iterations are shown in Figure 1 (b)–(d). In the third iteration (Figure 2(c)), the structure of the data is observable. In the 10th iteration (Figure 2(d)), the structure is even more clear. Finally, in the 20th iteration, (Figure 2(f)), the clusters are completely separated.

After the graph evolution iterations, the cluster structure encoded in the edge weight matrix is usually obvious to human. In practice, the number of clusters discovered by the algorithm is different from expected number of clusters. We use the following partition scheme to reach a desired number of cluster. We run algorithm in Eq. (6) until there are two disconnected subgraphs. Then pick up the subgraph which has a large number nodes to run algorithm in Eq. (6), and do the same strategy until we reach a specified number of clusters.

For the purpose of semi-supervised learning, we just use $\tilde{W} =$ **Graph Evolution**(W) as preprocessing, where W is the input of and $\tilde{W}$ is the output. Instead of performing semi-supervised learning on W, we do it on $\tilde{W}$. We show that the qualities of the $\tilde{W}$ are much higher than W.

4 Experimental Results

In this section, we first demonstrate the convergence of algorithm and then show experimental evidence of the quality improvement by apply our graph evolution algorithm. In the clustering comparison, we specify the number of clusters. However, in a microRNA pattern discovery application, we run our algorithm until convergence and let the algorithm determine the number of clusters.

4.1 Convergence Analysis

We first demonstrate the convergence of our algorithm on a toy data, which is a 9×9 binary graph, shown in the left most panel of the bottom row of Figure 3. There are two cliques in this graph: nodes 1–4 and nodes 5–9. We add some noise by setting $W_{13} = W_{58} = W_{79} = 0$ and $W_{45} = 1$. We run algorithm in Eq. (6) for 30 iterations. One can observe that our algorithm converges fast and at the convergent graph, all edges within the same clique have the same value. Also as highlighted in Figure 3, the noise values of W_{13}, W_{58}, W_{79}, and W_{45} are corrected by our algorithm.

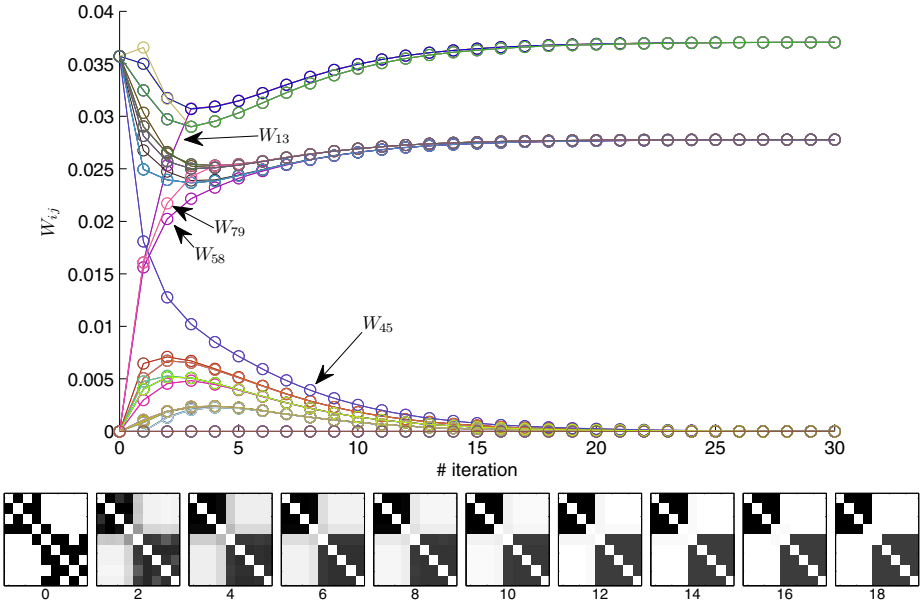


Fig. 3. Convergence curves and adjacency matrix of our algorithm on a 9×9 toy data. The left most panel of the bottom row is the initial binary graph (black represents 1 and white represents 0) and the rest of the bottom row is the evolution result of 2nd, 4th, $\cdots$, 18th iterations. Initially, nodes 1–4 is a pseudo-clique, as well as nodes 5–9. $W_{13} = W_{58} = W_{79} = 0$ and $W_{45} = 1$. After around 18 iterations, the two cliques become separated and the nodes within the two cliques become full connected. The top panel show the convergence of all the elements in W. Highlighted are the values of W_{13}, W_{58}, W_{79}, and W_{45}, which are corrected by our algorithm.

4.2 Clustering

In this experiment, we extensively compare our algorithm with standard clustering algorithms (K-means, Spectral Clustering, Normalized Cut[2]) in 20 data sets. These data sets come from a wide range of domains, including gene expressions including gene expressions (PR1,SRB, LEU, LUN, DER, AML, GLI, MAL, MLL), images (ORL, UMI, COI, JAF, MNI) and other standard UCI data sets (ION, PR2, SOY, ECO, GLA, YEA, ZOO, CAR, WIN, IRI)[3]. We use accuracy, normalized mutual information (NMI) and purity as the measurement of the clustering qualities and the results are shown in Table 1. Our method achieves the best results in 22 out of 24 data sets. Here notice that for Spectral Clustering and Normalized Cut, we tune the graph construction parameters. More

[2] We also compared with MCL. However the accuracies are much (more than 10%) lower than all the method we compare here. We believe MCL is not suitable for the purpose in this paper. One can find visual evidence in Figure 2.

[3] All the mentioned data can be downloaded at parchive.ics.uci.edu/ml/ or csie.ntu.edu.tw/ cjlin/.

Table 1. Accuracy, normalized mutual information (NMI), and purity comparison of K-mean (Km), Spectral Clustering (SC), Normalized Cut (Ncut), and Graph Evolution (GE). Both Spectral Clustering and Normalized Cut are achieved by tuning the graph construction parameters and the best results are reported.

	Accuracy				**NMI**				**Purity**			
	Km	SC	Ncut	GE	Km	SC	Ncut	GE	Km	SC	Ncut	GE
UMI	0.458	0.471	0.498	**0.644**	0.641	0.646	0.649	**0.763**	0.494	0.505	0.505	**0.667**
COI	0.570	0.614	0.792	**0.839**	0.734	0.750	0.860	**0.879**	0.623	0.658	0.817	**0.840**
ION	0.707	0.702	0.684	**0.880**	0.123	0.193	0.107	**0.446**	0.707	0.730	0.684	**0.880**
JAF	0.744	0.799	0.965	**0.967**	0.809	0.849	0.959	**0.962**	0.774	0.819	0.965	**0.967**
MNI	0.687	0.713	0.820	**0.833**	0.690	0.698	0.748	**0.769**	0.705	0.733	0.820	**0.833**
ORL	0.582	0.683	0.756	**0.775**	0.786	0.834	0.866	**0.891**	0.624	0.713	0.773	**0.802**
PR1	0.716	0.675	0.562	**0.899**	0.129	0.176	0.102	**0.458**	0.726	0.757	0.708	**0.899**
PR2	0.580	0.566	0.569	**0.706**	0.019	0.017	0.013	**0.136**	0.580	0.566	0.569	**0.706**
SOY	0.908	0.871	**1.000**	**1.000**	0.903	0.859	**1.000**	**1.000**	0.924	0.893	**1.000**	**1.000**
SRB	0.480	0.622	**0.699**	0.639	0.232	0.411	**0.454**	0.421	0.512	0.645	**0.699**	0.639
YEA	0.132	0.327	0.302	**0.395**	0.013	0.129	0.126	**0.231**	0.328	0.430	0.436	**0.540**
ZOO	0.264	0.674	0.629	**0.723**	0.116	0.615	0.570	**0.751**	0.423	0.750	0.737	**0.871**
AML	0.688	0.678	0.659	**0.847**	0.100	0.100	0.073	**0.394**	0.696	0.692	0.666	**0.847**
CAR	0.623	0.729	0.719	**0.799**	0.655	0.743	0.738	**0.779**	0.691	0.789	0.788	**0.822**
WIN	0.961	0.936	0.978	**0.983**	0.863	0.845	0.907	**0.926**	0.961	0.943	0.978	**0.983**
LEU	0.879	0.840	0.958	**0.972**	0.559	0.513	0.735	**0.806**	0.879	0.860	0.958	**0.972**
LUN	0.663	0.672	**0.748**	0.704	0.495	0.485	**0.547**	0.473	0.864	0.860	**0.911**	0.828
DER	0.766	0.848	0.955	**0.964**	0.838	0.818	0.905	**0.931**	0.853	0.876	0.955	**0.964**
ECO	0.552	0.496	0.505	**0.631**	0.467	0.458	0.487	**0.549**	0.739	0.770	0.808	**0.851**
GLA	0.452	0.446	0.453	**0.565**	0.320	0.298	0.333	**0.399**	0.549	0.572	**0.652**	0.650
GLI	0.585	0.548	0.559	**0.700**	0.465	0.410	0.398	**0.505**	0.619	0.569	0.601	**0.700**
IRI	0.802	0.746	0.843	**0.953**	0.640	0.514	0.655	**0.849**	0.815	0.758	0.843	**0.953**
MAL	0.911	0.731	0.902	**0.929**	0.569	0.299	0.544	**0.624**	0.911	0.743	0.902	**0.929**
MLL	0.669	0.637	0.687	**0.861**	0.435	0.376	0.426	**0.681**	0.692	0.651	0.687	**0.861**

explicitly the graph is constructed as $W_{ij} = \exp\left(-\|x_i - x_j\|^2/(\gamma \bar{r}^2)\right)$ where $\bar{r}$ denotes the average pairwise Euclidean distances among the data points and γ is chosen from $[2^{-2}, 2^{-1}, \cdots, 2^5]$ and the best results are reported.

4.3 Semi-supervised Learning

We first run graph evolution algorithm (Eq. (6)) for one iteration. After that we use the result weights as input to run Zhu *et al.*'s [19] (marked as HF in the Figure 5) and Zhou *et al.*'s [20] (marked as GC) approaches. We compare four methods, HF, GC, HF on resulting graph (HF_GE), GC on resulting graph (GC_GE), on four face image datasets. We tested the methods on AT&T[4], BinAlpha[5],

[4] http://people.cs.uchicago.edu/~dinoj/vis/ORL.zip
[5] http://www.cs.toronto.edu/~roweis/data.html

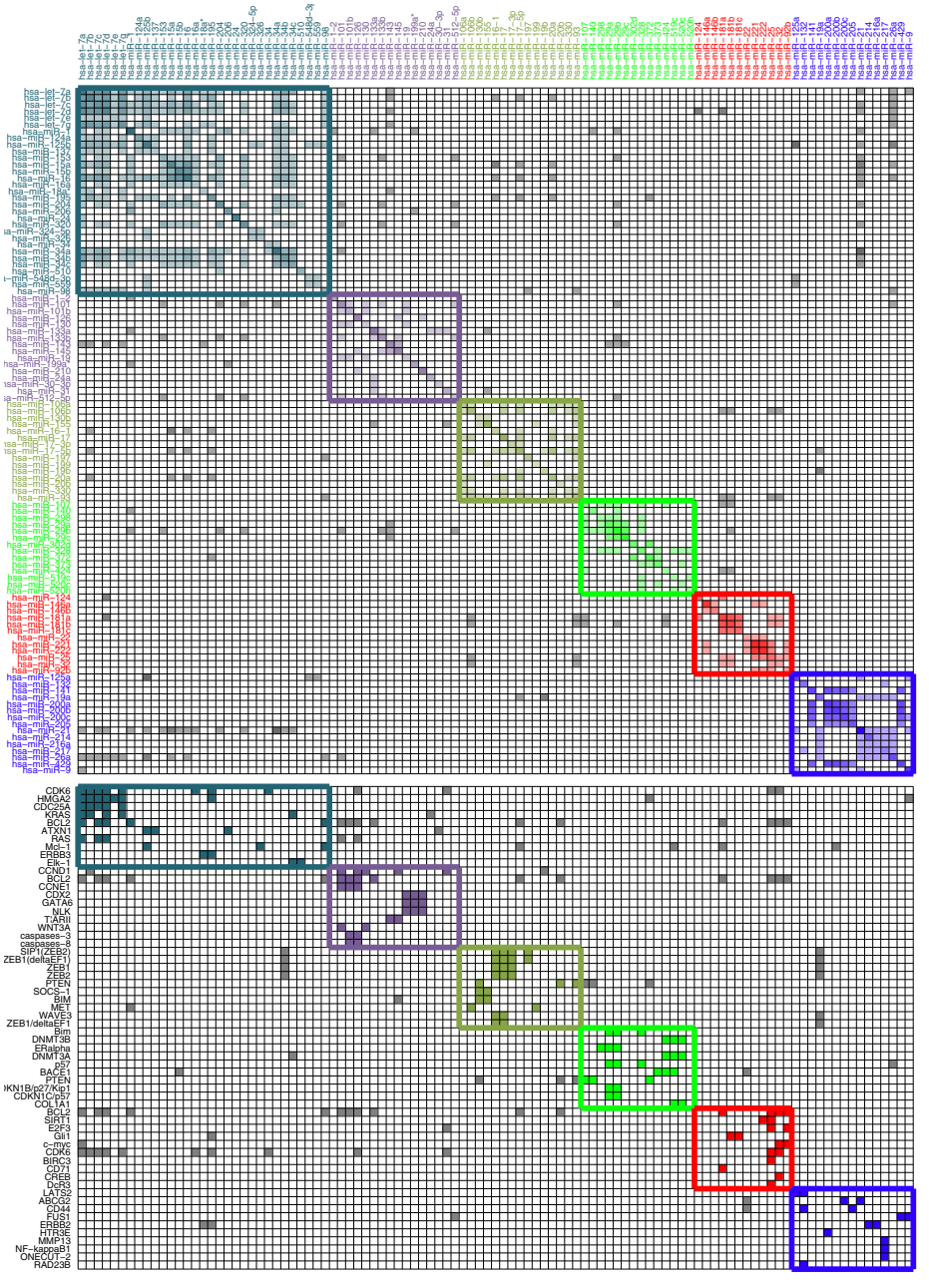


Fig. 4. 6 miRNA cliques found by Graph Evolution. Top panel is the miRNA graph in which the values denotes the number of common targeting genes of two miRNAs. The bottom panel is the top 10 targeting genes for each clique. The cliques are separated by different colors. The left top part of the top panel is the *let-7* miRNA family and the right bottom part of the top panel is the *hsa-mir-200* family.

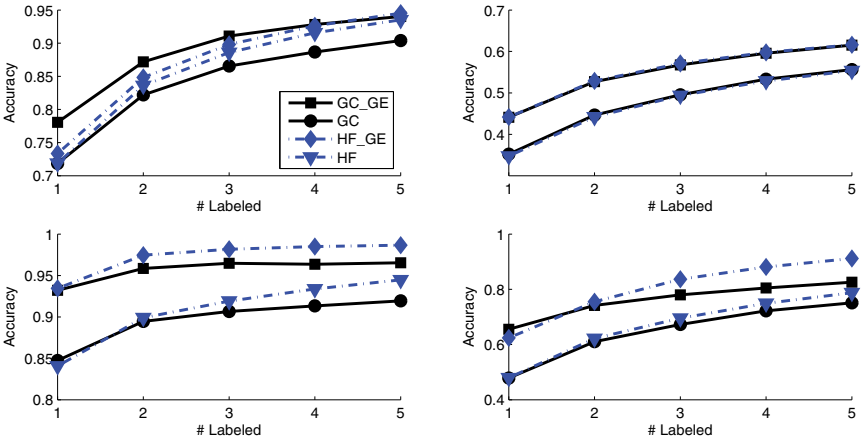


Fig. 5. Semi-supervised learning on 4 datasets(from left to right): AT&T, BinAlpha, JAFFE, and Sheffield datasets. Classification accuracies are shown for four methods: HF, GC, HF using condensated graph (HF_GE), GC using condensated graph (GC_GE). For each dataset, number of labeled data per class are set to 1, 2, 3, 4, 5. Using the graph evolution consistently improves over original methods.

JAFFE[6], and Sheffield[7] data sets. For all the methods and datasets, we randomly select N labeled images for each class, $N = 1, 2, 3, 4, 5$, and use the rest as unlabled images. We try 50 random selections for each dataset, and computer the average of the semi-supervised classification accuracy.

The results are shown in Figure 5. In all these case, we always obtain higher classification accuracy by applying graph condensation. For datasets BinAlpha, JAFFE, and Sheffield, our methods are consistently 5%–10% better than the standard semi-supervised learning methods.

4.4 Graph Evolution for microRNA Functionality Analysis

In this experiment, we are interested in the interaction network between microRNAs (miRNAs) and genes. MiRNAs play important regulatory roles by targeting messenger RNAs (mRNAs) for degradation or translational repression, and have become one of the focuses of post-transcriptional gene regulation in animals and plants [21,22,23] and have been an active research topic in various domains [24,25,26,27]. A database of verified miRNA/target gene relationship can be found in [28]. Here we apply our algorithm to investigate the relationships between the miRNAs and the genes. The main purpose is to discover new interaction patterns in the miRNA regulatory network.

We use the data with version of Nov. 6, 2010. We use the number of targeting genes as the weights of two miRNAs, *i.e.* $W_{ij} = \sum_k B_{ik}B_{jk}$ where $B_{ik} = 1$

[6] http://www.cs.toronto.edu/~roweis/data.html
[7] http://www.shef.ac.uk/eee/vie/face.tar.gz

indicates miRNA i targets gene k, $B_{ik} = 0$ otherwise. We select the largest disconnected component which has 103 miRNAs and run the **GraphEvolution** algorithm until converges. Finally, we discover 6 separated subgroups of miRNAs, which are shown in Figure 4. The following is the outline of our discovery in this experiment. (1) the *let-7* [29,30] miRNA family is correctly clustered into the same group. (2) The *hsa-mir-200* family are highly connected with each other, which is not reported in literature so far.

5 Conclusions

In this paper we present the Social Diffusion Process, which is motivated from the Matthew effect in social phenomenons. We develop the stochastic model by the assumption that social members tend to be together with someone who is familiar with. We also derive an graph evolution algorithm based on the presented mode. Empirical studies show significant improvement of the qualities of the graph data by the Social Diffusion Process, indicating that the assumptions in our model are natural in general. We also discover a new miRNA family in the experiment on miRNA functionality analysis.

Acknowledgment. This research is partially supported by NSF-CCF-0830780, NSF-DMS-0915228, NSF-CCF-0917274.

References

1. Kalton, A., Langley, P., Wagstaff, K., Yoo, J.: Generalized clustering, supervised learning, and data assignment. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 299–304. ACM, New York (2001)
2. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation 15, 1373–1396 (2003)
3. Shi, J., Malik, J.: Normalized cuts and image segmentation. In: IEEE Conf. Computer Vision and Pattern Recognition (1997)
4. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) NIPS, pp. 849–856. MIT Press, Cambridge (2001)
5. Chan, P.K., Schlag, M.D.F., Zien, J.Y.: Spectral K -way ratio-cut partitioning and clustering. In: DAC, pp. 749–754 (1993)
6. Hagen, L.W., Kahng, A.B.: New spectral methods for ratio cut partitioning and clustering. IEEE Trans. on CAD of Integrated Circuits and Systems 11, 1074–1085 (1992)
7. Nowicki, K., Snijders, T.A.B.: Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96, 1077–1088 (2001)
8. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels (2008)
9. Pitman, J.: Combinatorial stochastic processes. Technical report. Springer, New York (2002)

10. Blei, D., Griffiths, T., Jordan, M.: The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM 57, 21–30 (2010)
11. Blei, D., Frazier, P.: Distance dependent chinese restaurant processes. In: ICML (2010)
12. Jordan, M.I., Ghahramani, Z., Jaakkola, T., Saul, L.K.: An introduction to variational methods for graphical models. Machine Learning 37(2), 183–233 (1999)
13. Ishwaran, H., James, L.F.: Generalized weighted chinese restaurant processes for species sampling mixture models. Statistica Sinica 13, 1211–1236 (2003)
14. Ahmed, A., Xing, E.P.: Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In: SDM, pp. 219–230. SIAM, Philadelphia (2008)
15. van Dongen, S.: Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht (2000)
16. Merton, R.: The matthew effect in science. Science 159, 56–63 (1968)
17. Rossiter, M.W.: The matthew matilda effect in science. Social Studies of Science 23, 325–341 (1993)
18. Stanovich, K.E.: Matthew effects in reading: Some consequences of individual differences in the acquisition of literacy. Reading Research Quarterly 21, 360–407 (1986)
19. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: ICML, pp. 912–919 (2003)
20. Zhou, D., Bousquet, O., Lal, T., Weston, J., Scholkopf, B.: Learning with local and global consistency. In: NIPS, p. 321 (2003)
21. Bartel, D.P.: Micrornas: target recognition and regulatory functions. Cell 136, 215–233 (2009)
22. Bearfoot, J.L., et al.: Genetic analysis of cancer-implicated microrna in ovarian cancer. Clinical cancer research: an official Journal of the American Association for Cancer Research 14(22), 7246–7250 (2008)
23. Blenkiron, C., et al.: Microrna expression profiling of human breast cancer identifies new markers of tumor subtype. Genome Biology 8, R214+ (2007)
24. Ng, K., Mishra, S.: De novo svm classification of precursor micrornas from genomic pseudo hairpins using global and intrinsic folding measures. Bioinformatics 23, 1321–1330 (2007)
25. Huang, J.C., Frey, B.J., Morris, Q.: Comparing sequence and expression for predicting microRNA targets using genMIR3. In: Altman, R.B., Dunker, A.K., Hunter, L., Murray, T., Klein, T.E. (eds.) Pacific Symposium on Biocomputing, pp. 52–63. World Scientific, Singapore (2008)
26. Dahiya, N., Morin, P.: Micrornas in ovarian carcinomas. Endocrine-Related Cancer (2009) (in press), doi:10.1677/ERC–09–0203
27. Berezikov, E., et al.: Approaches to microrna discovery. Nat. Genet. 38(suppl.), S2–S7 (2006)
28. Jiang, Q., et al.: mir2disease: a manually curated database for microrna deregulation in human disease. Nucleic Acids Res. 37, D98–D104 (2009)
29. Hu, G., et al.: Microrna-98 and let-7 confer cholangiocyte expression of cytokine-inducible src homology 2-containing protein in response to microbial challenge. J. Immunol. 183, 1617–1624 (2009)
30. Abbott, A., et al.: The let-7 microrna family members mir-48, mir-84, and mir-241 function together to regulate developmental timing in caenorhabditis elegans. Dev. Cell 9, 403–414 (2005)

Multi-Subspace Representation and Discovery

Dijun Luo, Feiping Nie, Chris Ding, and Heng Huang

Department of Computer Science and Engineering,
University of Texas, Arlington, Texas, USA
{dijun.luo,feipingnie}@gmail.com, {chqding,heng}@uta.edu

Abstract. This paper presents the multi-subspace discovery problem and provides a theoretical solution which is guaranteed to recover the number of subspaces, the dimensions of each subspace, and the members of data points of each subspace simultaneously. We further propose a data representation model to handle noisy real world data. We develop a novel optimization approach to learn the presented model which is guaranteed to converge to global optimizers. As applications of our models, we first apply our solutions as preprocessing in a series of machine learning problems, including clustering, classification, and semi-supervised learning. We found that our method automatically obtains robust data presentation which preserves the affine subspace structures of high dimensional data and generate more accurate results in the learning tasks. We also establish a robust standalone classifier which directly utilizes our sparse and low rank representation model. Experimental results indicate our methods improve the quality of data by preprocessing and the standalone classifier outperforms some state-of-the-art learning approaches.

1 Introduction

The linear sparse representation approaches recently attract attentions from the researchers in statistics and machine learning. By providing robustness, simpleness, and sound theoretical foundations, sparse representation models have been widely considered in various applications [1,2,3,4].

In most previous models, we impose on the data an assumption that the data points can be linearly represented by other data points in the same class or data points nearby. This assumption will further lead to another assumption that subspace of each class has to include the original point. Our major argument in this paper is that this assumption is too loose in real world applications. For this reason, we further impose the affine properties of the subspaces and present a challenging affine subspace discovery problem. To be more specific, given a set of data points, which lie on multiple unknown spaces, we want to recover the membership of data points to subspaces, *i.e.* which data point belongs to which subspace. The major challenge here is that not only the subspaces and membership are unknown, but also the number of subspaces and the dimensions of the subspaces are unknown.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 405–420, 2011.

In this paper we will (1) present a sparse representation learning model to obtain the solutions automatically, which is theoretically guaranteed to recover all the unknown information listed above, (2) extended our model to handle noisy data and apply the sparse representation as a preprocessing in various machine learning tasks, such as unsupervised learning, classification and semi-supervised learning, and (3) develop a standalone classifier directly based on the sparse representation model. To handle the noisy data with robust performance, we introduce a mixed-norm optimization problem which involves trace, ℓ_2/ℓ_1, and ℓ_1 norms. We further develop an efficient algorithm to optimize the induced problem which is guaranteed to converge to a global optimizer.

Our model explicitly imposes both sparse and low rank requirements on the data presentation. We apply our model as preprocessing in various machine learning applications. The extensive and sound empirical results suggest that one might benefit from taking sparsity and low rank into consideration simultaneously.

2 Problem Description and Our Solution

Consider K groups data points $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K]$ and assume that there are $n_1, n_2, \cdots, n_K$ data points in each group, respectively ($\sum_{k=1}^{K} n_k = n$). We assume that for each group, the data points belong to independent affine subspaces. And the dimensions of the affine subspaces are $d_1, d_2, \cdots, d_K$. To be more specific, for each affine subspace $\mathbf{X}_k$, there exist $d_k + 1$ bases $\mathbf{U}^k = [\mathbf{u}_1^k, \mathbf{u}_2^k, \cdots, \mathbf{u}_{d_k}^k, \mathbf{u}_{d_k+1}^k]$ and for each data point $\mathbf{x} \in \mathbf{X}_k$, there exists $\boldsymbol{\beta}$ such that $\mathbf{x} = \mathbf{U}^k \boldsymbol{\beta}^k$ and that $\boldsymbol{\beta}^T \mathbf{1} = 1$. In this paper, by the dimension of the affine subspace, we mean the characteristic dimension, *i.e.* from the manifold point of view. Even though there are $d_k + 1$ bases in $\mathbf{U}^k$, we still consider that $\mathbf{U}^k$ defines a d_k-dimensional affine subspace.

2.1 Multi-Subspace Discovery Problem

The problem of **Multi-Subspace Discovery** is given $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K]$ to recover (1) the number of affine space K, (2) the dimension of each subspace d_k, and (3) the membership of the data points to the affine subspaces. The challenge in this problem is that the only known information is the input $\mathbf{X}$, where the data points are typically disordered, and all other information is unknown.

Will illustrate the Multi-Subspace Discovery problem in Figure 1. In this paper, we first derive a solution of this problem and provide several theoretical analysis of our solution on non-noisy data, then extend our model to handle noisy real-world case by adding ℓ_2/ℓ_1 norms which are convex but non-smooth regularizations. We develop an efficient algorithm to solve the problem.

2.2 A Constructive Solution

We cast the multi-subspace discovery problem into a trace norm optimization, in which the optimizer directly gives the number of affine subspace and the membership of the clustering. The results are theoretically guaranteed.

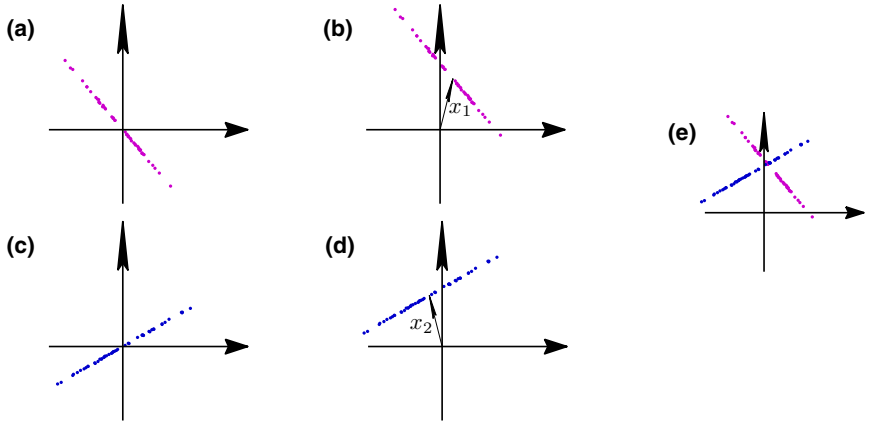


Fig. 1. A demonstration of the Multi-Subspace Discovery problem. **(a)** and **(c)**: Two groups of data points lying on two 1-dimension subspaces. **(b)**: All data points shifted by x_1 from **(a)**. **(d)**: All data points shifted by x_2 from **(c)**. **(e)**: A mixture of data points from **(b)** and **(d)**. The affine subspace clustering problem is to recover the number of subspaces (2 in this case), the membership of the data points to the subspaces (indicated by the color of the data points in **(e)**, the dimensions of the subspaces (1 for both of the subspace in this cases).

Representation of One Subspace

In order to introduce our solution in a more interpretable way, we first solve a simple problem in which there is only one affine subspace. Let $\mathbf{X}_1 = (\mathbf{x}_1, \cdots, \mathbf{x}_{n_1})$ be in a d_1-dimensional affine subspace spanned by the basis $\mathbf{U}_1$, $d_1 + 1 < n_1$, *i.e.* for each data points $\mathbf{x}_i$, there exists $\boldsymbol{\alpha}_i$,

$$\mathbf{x}_i = \mathbf{U}_1 \boldsymbol{\alpha}_i,\ \boldsymbol{\alpha}_i \in \mathbb{R}^{d_1+1},\ \boldsymbol{\alpha}_i^T \mathbf{1} = 1,\ 1 \leq i \leq n_1 \tag{1}$$

or more compactly, $\mathbf{X}_1 = \mathbf{U}_1 \mathbf{A},\ \mathbf{A}^T \mathbf{1} = \mathbf{1},$ where $\mathbf{1}$ is a column vector with all elements one in proper size and $\mathbf{A} = (\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_{n_1})$. We define

$$\tilde{\mathbf{X}}_1 = \begin{pmatrix} \mathbf{U}_1 \mathbf{A} \\ \mathbf{1}^T \end{pmatrix} \tag{2}$$

Then we have,

Lemma 1. *If* $\mathbf{X}_1$ *satisfies Eq. (1) and let*

$$\mathbf{Z}_1 = \tilde{\mathbf{X}}_1^+ \tilde{\mathbf{X}}_1 \tag{3}$$

where $\tilde{\mathbf{X}}_1$ *is defined in Eq. (2) and* $\tilde{\mathbf{X}}_1^+$ *is the* Moore-Penrose *pseudo inverse of* $\tilde{\mathbf{X}}_1$*, then*

$$\mathbf{X}_1 = \mathbf{X}_1 \mathbf{Z}_1,\ \mathbf{1}^T \mathbf{Z}_1 = \mathbf{1}^T, \tag{4}$$

and rank$(\mathbf{Z}_1) = d_1 + 1$.

Proof. By making use of the property of *Moore-Penrose* pseudo inverse, we immediately have

$$\tilde{\mathbf{X}}_1 = \tilde{\mathbf{X}}_1 \tilde{\mathbf{X}}_1^+ \tilde{\mathbf{X}}_1,$$

Thus,

$$\begin{pmatrix} \mathbf{U}_1\mathbf{A} \\ \mathbf{1}^T \end{pmatrix} = \begin{pmatrix} \mathbf{U}_1\mathbf{A} \\ \mathbf{1}^T \end{pmatrix} \mathbf{Z},$$

which is equivalent to two equations of

$$\mathbf{X}_1 = \mathbf{X}_1 \mathbf{Z}_1,$$

$$\mathbf{1}^T \mathbf{Z}_1 = \mathbf{1}^T.$$

It is obvious that $\text{rank}(\mathbf{Z}_1) = \text{rank}(\tilde{\mathbf{X}}_1)$. On the other hand, by the definition of $\mathbf{A}$ in Eq. (2), we have $\mathbf{1}^T\mathbf{A} = \mathbf{1}^T$, thus

$$\tilde{\mathbf{X}}_1 = \begin{pmatrix} \mathbf{U}_1\mathbf{A} \\ \mathbf{1}^T \end{pmatrix} = \begin{pmatrix} \mathbf{U}_1\mathbf{A} \\ \mathbf{1}^T\mathbf{A} \end{pmatrix} = \begin{pmatrix} \mathbf{U}_1 \\ \mathbf{1}^T \end{pmatrix} \mathbf{A} \tag{5}$$

From Eq. (2) we have

$$\text{rank}(\tilde{\mathbf{X}}_1) \geq \text{rank}(\mathbf{U}_1\mathbf{A}) = \text{rank}(\mathbf{X}_1) = d_1 + 1$$

But from Eq. (5) we have

$$\text{rank}(\tilde{\mathbf{X}}_1) \leq \text{rank}(\mathbf{A}) = d_1 + 1.$$

Thus $\text{rank}(\mathbf{Z}_1) = \text{rank}(\tilde{\mathbf{X}}_1) = d_1 + 1$.

Since $d_1 + 1 < n_1$, $\mathbf{Z}_1$ is low rank. Interestingly, this low-rank affine subspace presentation of Eqs. (1, 4) can be reformulated as a trace norm optimization problem:

$$\begin{aligned} &\min_{\mathbf{Z}_1} \|\mathbf{Z}_1\|_*, \\ &\text{s.t.} \;\; \mathbf{X}_1 = \mathbf{X}_1\mathbf{Z}_1, \; \mathbf{1}^T\mathbf{Z}_1 = \mathbf{1}^T \end{aligned} \tag{6}$$

where $\|\mathbf{Z}_1\|_*$ is the trace norm of $\mathbf{Z}_1$, *i.e.* the sum of singular values, or explicitly,

Lemma 2. $\mathbf{Z}_1$ *defined in Eq. (3) is an optimizer of the problem in Eq. (6).*

Due to the limited space, we omit the proof here[1].

In this paper, we hope to recover multiple $\mathbf{Z}$ which has diagonal block structure from $\mathbf{X}$ by which we solve the multi-subspace discovery problem.

Constructive Representation of K Subspaces

[1] One can also easily show that $\mathbf{Z}_1$ defined in Eq. (3) is one element in the subgradient of the Lagrangian $\mathcal{L}(\mathbf{Z}, \Lambda) = \|\mathbf{Z}\|_* - \mathbf{tr}(\tilde{\mathbf{X}}_1 - \tilde{\mathbf{X}}_1\mathbf{Z})^T\Lambda$,

Now consider the full case where the data points $\mathbf{X}$ belong exactly to K independent subspaces. Assume data points within a subspace are indexed sequentially, $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K]$. Repeat the above analysis for each subspace, we have

$$\mathbf{X} = [\mathbf{X}_1, \cdots, \mathbf{X}_K] = [\mathbf{X}_1\mathbf{Z}_1, \cdots, \mathbf{X}_K\mathbf{Z}_K] = \mathbf{XZ}, \tag{7}$$

where

$$\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{Z}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \mathbf{Z}_K \end{pmatrix} \tag{8}$$

Thus by construction, we have the following,

Theorem 1. *If* $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ *belong exactly to* K *subspaces of rank* d_k *respectively, there exists* $\mathbf{Z}$*, such that*

$$\mathbf{X} = \mathbf{XZ},\ \mathbf{1}^T\mathbf{Z} = \mathbf{1}^T. \tag{9}$$

where $\mathbf{Z}$ *has the structure of Eq.(8) and rank*$(\mathbf{Z}_k) = d_k + 1, 1 \leq k \leq K$.

Recovery of The Multiple Subspaces

Intuited by Lemma 2, and Theorem 1, one might hypothetically consider recovering the block structure by using the following optimization,

$$\begin{aligned} &\min_{\mathbf{Z}} \|\mathbf{Z}\|_*, \\ \text{s.t.}\ \ &\mathbf{X} = \mathbf{XZ},\ \mathbf{1}^T\mathbf{Z} = \mathbf{1}^T, \end{aligned} \tag{10}$$

which is a convex problem since the objective function $\|\mathbf{Z}\|_*$ is a convex function *w.r.t* $\mathbf{Z}$ and the domain constraints $\mathbf{X} = \mathbf{XZ},\ \mathbf{1}^T\mathbf{Z}_1 = \mathbf{1}^T$ is an affine space, which is a convex domain. This is desirable property: if a solution $\mathbf{Z}^*$ is a local solution, $\mathbf{Z}^*$ must be a global solution. However, a convex optimization could have multiple global solutions, *i.e.*, the global solution is not unique.

This optimization indeed has one optimal solution:

Theorem 2. *The optimization problem of Eq. (10) has the optimal solution*

$$\mathbf{Z}^* = \tilde{\mathbf{X}}^+\tilde{\mathbf{X}} \tag{11}$$

where

$$\tilde{\mathbf{X}} = \begin{pmatrix} \mathbf{X} \\ \mathbf{1}^T \end{pmatrix}. \tag{12}$$

In general, $\mathbf{Z}^*$ is not sparse and does not have the sparse block structure of $\mathbf{Z}$ in Eq. (8). Similar data representation model was represented in [5], which suffers from the same problem. Here we extend the model to solve the general multi-subspace problem and provide a proof of the uniqueness of the solution.

To recover a solution which has the sparse structure of Eq. (8), we add a ℓ_1 term to optimization Eq. (10) to promote sparsity of the solution, and optimize the following

$$\begin{aligned} &\min_{\mathbf{Z}} J_1(\mathbf{Z}) = \|\mathbf{Z}\|_* + \delta\|\mathbf{Z}\|_1 \\ &\text{s.t. } \mathbf{X} = \mathbf{X}\mathbf{Z},\ \mathbf{1}^T\mathbf{Z}_1 = \mathbf{1}^T, \end{aligned} \tag{13}$$

where $\|\mathbf{Z}\|_1$ is the element-wise ℓ_1 norm: $\|\mathbf{Z}\|_1 = \sum_{ij} |Z_{ij}|$ and δ is model parameter which control the balance between low rank and sparsity. In our theoretical studies, we only require $\delta > 0$. Because the ℓ_1 norm is convex and the optimization problem (13) is strictly convex at the minimizer, it has unique solution.

And fortunately, for problem Eq.(13), we have the following theorem,

Proposition 1. *Assume $\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K$ are independent affine subspaces. Let $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K]$, then all the minimizers of problem Eq.(13) have the form of Eq.(8). Further more, each block $\mathbf{Z}_k$ has only one connected component.*

The proof of Proposition 1 can be found in the supplementary materials of this paper.

Since each block $\mathbf{Z}_k$ has only one connected component and all the whole $\mathbf{Z}$ is block diagonal, the number of affine subspaces is trivial to recovered, which is the number of connected components of $\mathbf{Z}$. The membership of each data points to the affine spaces is also guaranteed to be recovered.

3 Multi-Subspace Representation with Noise

Typically data are drawn from multiple subspaces but with noise. Thus $\mathbf{X} = \mathbf{X}\mathbf{Z}$ does not hold anymore for any low rank $\mathbf{Z}$. On the other hand, we can combine the two constraints in Eq. (13) as,

$$\begin{pmatrix} \mathbf{X} \\ \mathbf{1}^T \end{pmatrix} = \begin{pmatrix} \mathbf{X} \\ \mathbf{1}^T \end{pmatrix} \mathbf{Z}. \tag{14}$$

With the notation of $\tilde{\mathbf{X}}$ in Eq. (12), we have $\tilde{\mathbf{X}} = \tilde{\mathbf{X}}\mathbf{Z}$. We may express the relationship as $\tilde{\mathbf{X}} = \tilde{\mathbf{X}}\mathbf{Z} + \mathbf{E}$, where $\mathbf{E}$ represents noise. To handle such noise case, in the optimization objective of Eq.(13), we add the term

$$\|\mathbf{E}\|_{\ell_2/\ell_1} = \sum_j \sqrt{\sum_i \mathbf{E}_{ij}^2} = \sum_{j=1}^{n} \left\| \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix} - \begin{pmatrix} \mathbf{X} \\ \mathbf{1}^T \end{pmatrix} \mathbf{z}_j \right\|.$$

This is the ℓ_2/ℓ_1-norm of matrix of $\mathbf{E}$. This norm is more robust against outliers than the usual Frobenius norm. With this noise correction term, we solve,

$$\min_{\mathbf{Z}} \ \|\tilde{\mathbf{X}} - \tilde{\mathbf{X}}\mathbf{Z}\|_{\ell_2/\ell_1} + \lambda\|\mathbf{Z}\|_* + \delta\|\mathbf{Z}\|_1, \tag{15}$$

where λ and δ are parameters which control the importance of $\|\mathbf{Z}\|_*$ and $\mathbf{Z}_1$, respectively.

3.1 Multi-Subspace Representation

Notice that if the data contain noise and the constraints in Proposition 1 do not hold, we lose the guarantee of the block diagonal structure of $\mathbf{Z}$. However, since the low rank and sparsity regularizer of Eq. (15), the final solution $\mathbf{Z}$ can be interpreted as representation coefficient of $\mathbf{X}$. We call such representation as Multi-Subspace Representation (MSR).

In summary, MSR representation of data $\mathbf{X}$ is given by the following:
(1) From input data $\mathbf{X}$, solve the optimization Eq.(15) to obtain $\mathbf{Z}$;
(2) The MSR representation of $\mathbf{X}$ is $\mathbf{XZ}$, i.e., the representation of $\mathbf{x}_i$ is $\mathbf{Xz}_i$.

In §4, we develop an algorithm to solve Eq. (15) and in §5, some applications of our model in machine learning are given.

3.2 Relation to Previous Work

The MSR representation here is motivated by the affine subspace clustering problem. However, some properties of the representation have been investigated in previous work by other researchers. First notice that $\mathbf{Z}$ is sparse, the representation of $\mathbf{x}_i \approx \mathbf{Zz}_i$ is similar to the one in sparse coding [6,7]. Interestingly, research in other communities suggests that in the natural process and even in human cognition, information is often organized in a sparse way, *e.g.* Vinge *et al.* discover that primary visual cortex (area *V1*) uses a sparse code to efficiently represent natural scenes [8].

In the sparse representation model, for each testing object, we seek a sparse representation of the testing object by all objects in training data set. Such learning mechanisms implicitly learn the structure, under the assumption that the sparse representation coefficients are imbalanced among groups. To be more specific, given a set of training data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ ($p \times n$ matrix, where p is the dimension of the data) and a testing data point $\mathbf{x}_t$, they solve the following optimization problem

$$\min_{\alpha_t} \|\mathbf{x}_t - \mathbf{X}\alpha_t\|^2 + \lambda\|\alpha_t\|_1, \tag{16}$$

where α_t ($n \times 1$ vector) has the reconstruction coefficients of x_t using all the training data objects $\mathbf{X}$, λ is the model parameter, and $\|\cdot\|_1$ is the ℓ_1 norm: $\|a\|_1 = \sum_{i=1} |a_i|$.

Wright *et al* introduce the Sparse Represented-Based Classification method [9], which uses the following strategy for class prediction,

$$\arg\min_k r_k = \|\mathbf{x}_t - \mathbf{X}\alpha_t^k\|, \tag{17}$$

where r_k is the representation error using the training samples in group k and α_t^k is obtained by setting the coefficients in α_t, corresponding to training samples not in class k, to zero, *i.e.*

$$\alpha_t^k(i) = \begin{cases} \alpha_t(i) & \text{if } i \in C_k, \\ 0 & \text{otherwise,} \end{cases}$$

where C_k is a set of all data points in class $k, k = 1, 2, \cdots, K$, and K is the number of classes.

On the other hand, $\mathbf{Z}$ in our model is also low rank, which is a natural requirement of most of data representation techniques, such as the low rank kernel methods [10] and robust Principle Component Analysis [11]. One can easily find literacy of the low rank representation in real world applications in various domains which indicates that low rank is one of the intrinsic properties of the data we observe, *e.g.* the missing value recover of DNA microarrays [12].

By combining the two basic properties (sparsity and low rank), our model naturally captures a proper representation of the data. We will demonstrate the quality of such representation using comprehensive empirical evidences in the experimental section.

4 An Efficient Algorithm and Analysis

4.1 Outline of the Algorithm

Assume we are solving a general problem of

$$J(\mathbf{x}) = f(\mathbf{x}) + \phi(\mathbf{x}), \tag{18}$$

where $f(\mathbf{x})$ is smooth and $\phi(\mathbf{x})$ is non-smooth and convex. If one of the elements in subgradient of $\phi(\mathbf{x})$ can be written as product of $g(\mathbf{x})$ and $h(\mathbf{x})$, *i.e.*,

$$g(\mathbf{x})h(\mathbf{x}) \in \partial\phi(\mathbf{x}),$$

where $h(\mathbf{x})$ is smooth and $\partial\phi(\mathbf{x})$ is the subgradient of $\phi(x)$, then instead of solving Eq. (18), we iteratively solve the following,

$$\mathbf{x}^{t+1} = \arg\min_{\mathbf{x}} \tilde{J}(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}^t) \int h(\mathbf{x}) d\mathbf{x}. \tag{19}$$

Notice that $\partial\tilde{J}(\mathbf{x})/\partial\mathbf{x} \in \partial J(\mathbf{x})$ when $\mathbf{x} = \mathbf{x}^t$. Hopefully, at convergence, $\mathbf{x}^{t+1} = \mathbf{x}^t$, then $\mathbf{0} \in \partial J(\mathbf{x})$ at $\mathbf{x}^t$, which means $\mathbf{x}^t$ is an optimizer of $J(\mathbf{x})$.

In general, the iterative steps in Eq. (19) cannot guarantee the convergence of $\mathbf{x}$ (*i.e.* $\mathbf{x}^{t+1} = \mathbf{x}^t$), and even the convergence of $J(\mathbf{x})$ (*i.e.* $J(\mathbf{x}^{t+1}) = J(\mathbf{x}^t)$). Fortunately, in our case of Eq. (15), our optimization technique guarantees both, and thus our algorithm guarantees to be an optimizer. Further more, in our algorithm, optimization problem in Eq. (19) has a close form solution, thus our algorithm is efficient.

4.2 Optimization Algorithm

Here we first present the optimization algorithm of Eq.(15), and then present theoretical analysis of the algorithm.

The algorithm is summarized in Algorithm 1. In the algorithm, $\mathbf{z}_i$ denotes the i-th column of $\mathbf{Z}$. The converged optimal solution is only weakly dependent on

Algorithm 1. $(\mathbf{X}, \lambda, \delta)$

Input: Data $\mathbf{X}$, model parameters λ, δ
Output: $\mathbf{Z}$ which optimizes Eq.(15).
Initialization: Compute $\tilde{\mathbf{X}}$ using Eq. (12), $\mathbf{Z} = \mathbf{0}$.
while not converged **do**
  $\mathbf{B} = \left(\mathbf{Z}\mathbf{Z}^T + \epsilon\mathbf{I}\right)^{-1/2}$
  for $i = 1 : n$ **do**
    $\mathbf{d}_i = \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}\mathbf{z}_i\|$,
    $\mathbf{D}_i = \mathbf{diag}\left(Z_{1i}^{-1}, Z_{2i}^{-1}, \cdots, Z_{ni}^{-1}\right)$,
    $\mathbf{z}_i = \left[\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} + \lambda\mathbf{d}_i\left(\mathbf{B} + \delta\mathbf{D}\right)\right]^{-1}\tilde{\mathbf{X}}^T\tilde{\mathbf{x}}_i$,
  end for
end while
Output: Z

parameter. We set δ to $\delta = 1$. ϵ is an auxiliary constant for improving numerical stability in computing trace norm. We set $\epsilon = 10^{-8}$ in all experiments.

In the third line of the **for** loop, we are actually solving the problem in Eq. (19). In practice, we do not explicitly compute the inverse. Instead, we solve the following linear equation to obtain $\mathbf{z}_i$,

$$\left[\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} + \lambda\mathbf{d}_i\left(\mathbf{B} + \delta\mathbf{D}\right)\right]\mathbf{z}_i = \tilde{\mathbf{X}}^T\tilde{\mathbf{x}}_i. \tag{20}$$

The algorithm is simple which involves no other optimization procedures. The algorithm generally converges in about 10 iterations in our experiments.

We have developed theoretical analysis for this algorithm, convering three properties for this algorithm: convergence, objective function value decreasing monotonically, and converging to global solution.

4.3 Theoretical Analysis of Algorithm 1

Before presenting the main theories for Algorithm 1, we first introduce two useful lemmas here.

Lemma 3

$$\|\mathbf{Z}\|_* = \lim_{\epsilon \to 0} \mathbf{tr}\left(\mathbf{Z}\mathbf{Z}^T + \epsilon\mathbf{I}\right)^{1/2}, \tag{21}$$

and

$$\lim_{\epsilon \to 0}\left(\mathbf{Z}\mathbf{Z}^T + \epsilon\mathbf{I}\right)^{-1/2}\mathbf{Z} \in \partial\|\mathbf{Z}\|_*, \tag{22}$$

where $\partial\|\mathbf{Z}\|_*$ *is the subgradient of trace norm.*

Here $\epsilon\mathbf{I}$ is introduced for numerical stability.

Lemma 4. *Assume matrices* $\mathbf{Z}$ *and* $\mathbf{Y}$ *have the same size. Let* $\mathbf{A} = \left(\mathbf{Y}\mathbf{Y}^T + \epsilon\mathbf{I}\right)^{1/2}$ *and* $\mathbf{B} = \left(\mathbf{Z}\mathbf{Z}^T + \epsilon\mathbf{I}\right)^{1/2}$. *Then the following holds*

$$\mathbf{tr}\mathbf{A} - \mathbf{tr}\mathbf{B} + \frac{1}{2}\mathbf{tr}\mathbf{Z}^T\mathbf{B}^{-1}\mathbf{Z} - \frac{1}{2}\mathbf{tr}\mathbf{Y}^T\mathbf{B}^{-1}\mathbf{Y} \leq 0. \tag{23}$$

Proof

$$\begin{aligned}
&\mathbf{trA} - \mathbf{trB} + \frac{1}{2}\mathbf{trZ}^T\mathbf{B}^{-1}\mathbf{Z} - \frac{1}{2}\mathbf{trY}^T\mathbf{B}^{-1}\mathbf{Y} \\
=&\mathbf{trA} - \mathbf{trB} + \frac{1}{2}\mathbf{trB}^{-1}\left(\mathbf{ZZ}^T - \mathbf{YY}^T\right) \\
=&\frac{1}{2}\mathbf{trB}^{-1}\left(2\mathbf{BA} - 2\mathbf{B}^2 + \mathbf{ZZ}^T - \mathbf{YY}^T\right) \\
=&\frac{1}{2}\mathbf{trB}^{-1}\left(2\mathbf{BA} - 2\mathbf{B}^2 + \mathbf{ZZ}^T + \epsilon\mathbf{I} - \mathbf{YY}^T - \epsilon\mathbf{I}\right) \\
=&\frac{1}{2}\mathbf{trB}^{-1}\left(2\mathbf{BA} - \mathbf{B}^2 - \mathbf{A}^2\right) \\
=&-\frac{1}{2}\mathbf{trB}^{-1/2}\left(\mathbf{A} - \mathbf{B}\right)^2\mathbf{B}^{-1/2} \leq 0.
\end{aligned}$$

One should notice that here $\mathbf{A}$ and $\mathbf{B}$ are symmetric full rank matrices.

Lemma 4 serves as a crucial part of our main theorem, which is stated as follows,

Theorem 3. *Algorithm 1 monotonically decreases the following objective,*

$$\min_{\mathbf{Z}} J(\mathbf{Z}) = \|\tilde{\mathbf{X}} - \tilde{\mathbf{X}}\mathbf{Z}\|_{\ell_2/\ell_1} + \lambda\mathbf{tr}\left(\mathbf{ZZ}^T + \epsilon\mathbf{I}\right)^{\frac{1}{2}} + \delta\|\mathbf{Z}\|_1, \tag{24}$$

i.e. $J(\mathbf{Z}_{t+1}) \leq J(\mathbf{Z}_t)$, *where* $\mathbf{Z}_t$ *is the solution of* $\mathbf{Z}$ *in the* t*-th iteration.*

Since the objective in Eq.(24) is lower bounded by 0, Theorem 3 guarantees the convergence of the objective value. Further more, we have

And according to Lemma 3, we know that the above solution is also the optimal solution of Eq.(15) when $\epsilon \to 0$.

We provide the proofs of all the theoretical analysis above in the supplementary materials.

5 Applications

5.1 Using Multi-Subspace Representation as Preprocessing

Since $\mathbf{Z}$ is low rank, $\mathbf{XZ}$ is also low rank. And since $\mathbf{Z}$ is sparse, $\mathbf{XZ}$ can be interpreted as a sparse coding representation of $\mathbf{X}$. According to the analysis in §3.2, we hopefully improve the qualities of the data representation by using $\mathbf{XZ}$. In our study, we replace $\mathbf{X}$ by $\mathbf{XZ}$ as a preprocessing step for various machine learning problems, where $\mathbf{Z}$ is the optimal solution of Eq. (15).

Notice that the learning of $\mathbf{Z}$ in Eq. (15) is unsupervised, which requires no further label information. Thus we can apply it as preprocessing for any machine learning tasks, as long as the data are represented in Euclidean space. In this paper, we employ MSR for clustering, semi-supervised learning, and classification. We will demonstrate the performance of the preprocessing in the experimental section.

5.2 Using Multi-Subspace Representation as Classifier

Here we try to directly make use of our MSR model as a standalone classifier. Assume we have n data points in the data set, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ and the first m data points have discrete class labels $y_1, y_2, \cdots, y_m$ in K classes, $y_i \in \{1, 2, \cdots, K\}$. The classification problem is to determine the class label of $\mathbf{x}_i, i = m+1, \cdots, n$. Let $\mathbf{Z}$ be the optimal solution of Eq.(15) for n data points. The MSR representation of each image is $\mathbf{X}\mathbf{z}_i,\ i = 1, \cdots, n$. The class prediction of our model for unlabeled data $\mathbf{x}_t,\ t = m+1, \cdots, n$, is

$$\arg\min_k r_k = \|\mathbf{X}\mathbf{z}_t - \hat{\mathbf{x}}_t^k\|,\ \hat{\mathbf{x}}_t^k = \sum_{i \in C_k} \mathbf{x}_i Z_{it}. \tag{25}$$

Here $\hat{\mathbf{x}}_t^k$ is the representation of testing object $\mathbf{x}_t$ using objects in class C_k, $k = 1, 2, \cdots, K$.

The classification strategy is similar with Wright *et al*'s approach [9]. We will compare the two models in the experimental section.

6 Experiment

6.1 A Toy Example

We demonstrate with toy example of the affine space recovering by our method in Figure 2. **(a)** shows 100 images from 10 groups used in this example, which are selected from the AT&T data set, details can be found in the experimental section. In order to obtain 10 affine subspaces which satisfy the constraints in Proposition 1, we remove the last principle component in each group of face images. To be more specific, for each group $\mathbf{X}_k$, we first subtract the data points by the group mean $\mathbf{m}_k : \bar{\mathbf{X}}_k = \mathbf{X}_k - \mathbf{m}_k \mathbf{1}^T$, then perform a PCA (Principle Component Analysis) on the zero-mean data and keep the first 8 principle components and get rid of the 9-th principle component. Then the data is projected back on to the original space and the mean $\mathbf{m}_k$ is added back. Assume the resulting PCA projection is $\mathbf{U}_k$ then the processed data $\mathbf{Y} = \mathbf{U}_k \mathbf{U}_k^T \bar{\mathbf{X}}_k + \mathbf{m}_k$ are used in our example, $k = 1, 2, \cdots, 10$. The images in which the last principle component have been removed are shown in Figure 2 (**a**). Notice that they are visually almost identical to the original image since the energy of the last component is close to zero. Then we solve Eq. (13) and the optimal solution is shown in Figure 2 (**b**), in which white color represents zeros, blue colors represent negative values, and red positive values. One can see that within each group, the values of the subgraph represented by $\mathbf{Z}_k$ (defined in Eq. (8)) is a single connected component and among the ten $\mathbf{Z}_k, k = 1, 2, \cdots, 10$ they are disconnected components.

As suggested in the previous section, our multi-subspace representation model has various potential real world applications. In the section, we will verify the quality of our model as a preprocessing method in three types of machine learning tasks, *i.e.* clustering, semi-supervised learning, and classification. We also evaluate our model as a standalone classifier.

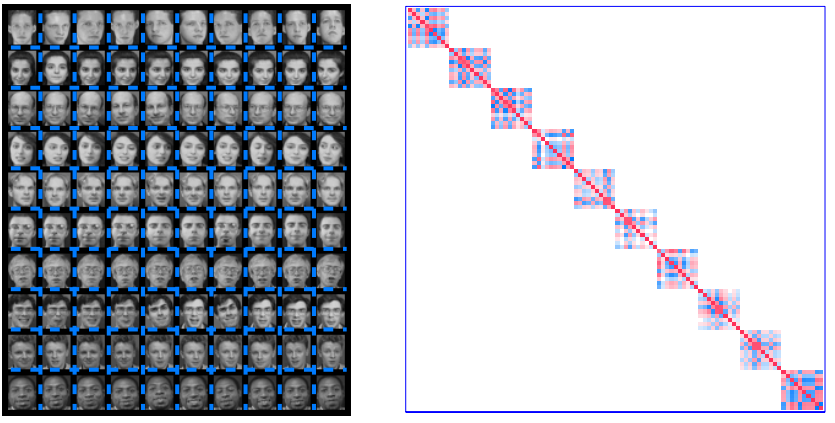


Fig. 2. A toy example of multi-subspace discovery problem and our solution. **(a)**: 100 images in which the last component has been removed within each group. Each row is one group which has 10 images. Within each group, the data are rank deficient, which satisfy the conditions in Proposition 1. **(b)**: the optimal solution of $\mathbf{Z}$ in Eq. (13). White color represents zeros, blue colors represent negative values, and red positive values. Within each group, the values of the subgraph represented by $\mathbf{Z}_k$ (defined in Eq. (8)) is a single connected component and the among the 10 $\mathbf{Z}_k, k = 1, 2, \cdots, 10$ they are disconnected components.

6.2 Experimental Settings

Datasets

We evaluate the performance of our model on 5 real world datasets, including two face image data bases, **LFW** (Labeled Faces in the Wild)[2], **AT&T**[3], two UCI datasets **Austrian** and **Dermatology** [13], and one handwritten character data BinAlpha[4]. All the data sets are used with the original data, without any further preprocessing.

Compared Methods

For the usage of preprocessing of our model, we compare 3 clustering algorithms (Normalized Cut [14], Spectral Embedding Clustering [15] and K-means), two standard semi-supervised learning algorithms (Local and Global Constancy by [16] and Gaussian Fields and Harmonic Functions by [17]), and two standard classification algorithms (linear Support Vector Machines and k-Nearest Neighbor).

For the usage of standalone classifier, we compare our method with Wright *et. al*'s sparse representation based approach [9].

[2] http://www.itee.uq.edu.au/∼conrad/lfwcrop/
[3] http://people.cs.uchicago.edu/˜dinoj/vis/ORL.zip
[4] http://www.cs.toronto.edu/˜roweis/data.html

Validation Settings
All the clustering algorithms compared in our experiments require random initializations. Thus we run the algorithms for 50 random trials and report the averages. For semi-supervised learning, we randomly split the data into 30% and 70% where the 30% of the data points are used as labeled data and 70% are used as unlabeled data. We repeat the random splitting for 50 times, where the average result is reported. For classification, when comparing our method as a preprocessing algorithm, we use the same splitting strategy as in semi-supervised learning, but splitting in to 50% for training and the other half for testing. For classification, when comparing our method as a standalone classifier, we use 30% for training and the rest 70% for testing. The reason is that for some of the datasets, the data points are well separated and the classification accuracy is very high, then the difference between approaches is not obvious. Thus here we use fewer data samples as the training set to enlarge the differences.

Parameter settings
K-means has no parameters. For kNN we use $k = 1$, *i.e.* just use the nearest neighbor classifier. For the Normalized Cut (NCut), Spectral Embedding Clustering (SEC) in clustering, Local and Global Constancy (LGC), and Gaussian Fields and Harmonic Functions (GFHF) in semi-supervised learning, we establish the graph using Gaussian kernel: $W_{ij} = \exp\left(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2\right)$, where γ is the parameter which is set to be $\gamma = [0.1, 0.5, 1, 2, \cdots, 30]$ and σ is the average of pairwise Euclidian distances among all data points.

For Wright *et. al*'s sparse representation (SR), we use LARS [18] to obtain the full LASSO path solution and use m top ranked coefficients according to the shrinking order in LARS solution path. We choose m from $m = 1, 2, \cdots \min(n, p)$ where n is the number of data points and p is the number of data dimension. The reason we use LARS is that it is more efficient than any other ℓ_1 solver in the sense that LARS computes all the possible solution with different parameters at once and for other solver, we need to retrain the model every time we change the parameter, which is time consuming for the purpose of highly parameter tuning. For our method, we choose λ from $[0.5, 0.6, \cdots, 2.5]$.

6.3 Experimental Results

For the usage of preprocessing our model, the results are shown in Figure 3. Here we show the average accuracies for both original data without processing (marked as **Orig** in the figure) and the corresponding method on the preprocessed data by our method (marked as **MSR**). We further plot the original accuracy values of all the 50 random trials for each methods to visualize the overall differences of the performance.

One-way ANOVA (Analysis of Variance) is performed to test how significantly our method is better than the original method, and corresponding p value is also shown in the figure. $p \leq \epsilon$ means p is less than any positive values in machine precision, *i.e.* the p value is very close to 0.

Out of the $5 \times 7 = 35$ comparisons, our method significantly outperforms the original methods in 33 comparisons, with $p \leq 0.03$. There is one case (SVM on

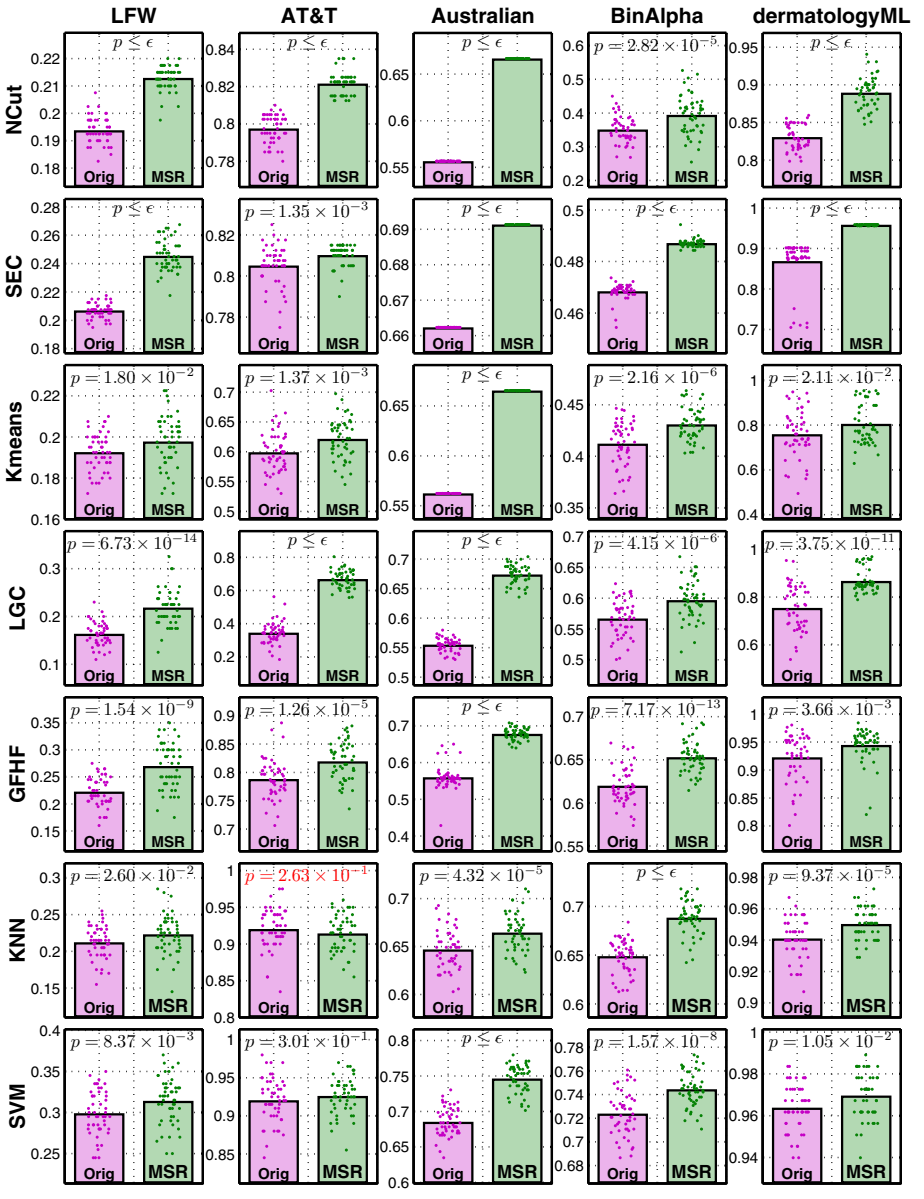


Fig. 3. Experimental results of our method as a preprocessing method on 7 learning methods and 5 data sets. The scattering dots represent the accuracy values of the methods and bars represent the averages. **Orig** and **MSR** denote the corresponding method on the original data and on the preprocessed by our method, respectively. The p stands for the significance of the one-way ANOVA test (for the hypothesis of "our method is better than the original method"). Out of 35 comparison, our method significantly outperforms the original methods in 33 cases, with $p \leq 0.03$. ϵ is the smallest positive values by machine precision.

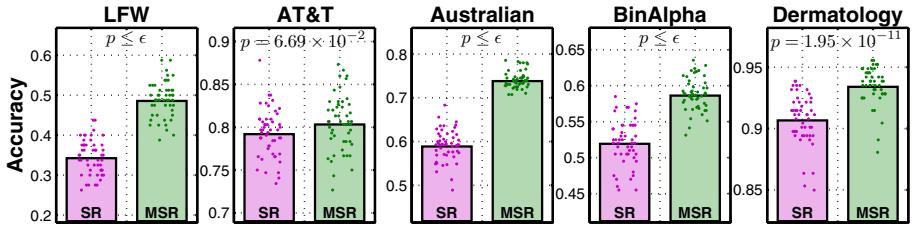


Fig. 4. A comparison of our model (**MSR**) and the Sparse Representation based method (**SR**) on 5 data sets. The p values represents the significance of one-way ANOVA test of the hypothesis "our method is better than SR".

AT&T data set) where our method is better but with no significant evidence. There is also another case in which our method is worse than the original method (kNN on **AT&T**), but the difference is not significant ($p = 0.263$).

For our model as a standalone classifier, the comparison results with Sparse Representation based method are shown in Figure 4. Out of 5 data sets, our method is significantly better than the Sparse Representation based method in four with $p \leq 0.01$.

7 Conclusions

In this paper, we present the multi-subspace representation and discovery model, which is motivated by the multi-subspace discovery problem. We solve the multi-subspace discovery problem by providing block diagonal representation matrix where the data points are connected in the same subspace and disconnected for different subspace. We then extend our approach to handle noisy real world data which leads to the Multi-Subspace Representation. We develop an efficient algorithm for the presented model and a global optimizer is guaranteed. Empirical studies suggest that our method improves the quality of the data by sparse and low rank representation and the induced standalong classifier outperforms standard sparse representation approach.

Acknowledgment. This research is partially supported by NSF-CCF-0830780, NSF-DMS-0915228, NSF-CCF-0917274.

References

1. Jenatton, R., Obozinski, G., Bach, F.: Structured sparse principal component analysis. In: Proc. AISTATS. Citeseer (2009)
2. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B 67, 301–320 (2005)
3. Beygelzimer, A., Kephart, J., Rish, I.: Evaluation of optimization methods for network bottleneck diagnosis. In: ICAC (2007)

4. Luo, D., Ding, C., Huang, H.: Towards structural sparsity: An explicit ℓ_2/ℓ_0 approach. In: 2010 IEEE International Conference on Data Mining, pp. 344–353. IEEE, Los Alamitos (2010)
5. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: Proceedings of the 26th International Conference on Machine Learning. Citeseer, Haifa (2010)
6. Olshausen, B.A., Field, D.J.: Sparse coding with an overcomplete basis set: A strategy employed by V1? Vision research 37(23), 3311–3325 (1997)
7. Tibshirani, R.: Regression shrinkage and selection via the LASSO. J. Royal. Statist. Soc. B 58, 267–288 (1996)
8. Vinje, W.E., Gallant, J.L.: Sparse coding and decorrelation in primary visual cortex during natural vision. Science 287, 1273 (2000)
9. Wright, J., Yang, A., Ganesh, A., Sastry, S., Ma, Y.: Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 210–227 (2009)
10. Bach, F., Jordan, M.: Predictive low-rank decomposition for kernel methods. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 33–40. ACM, New York (2005)
11. Candes, E., Li, X., Ma, Y., Wright, J.: Robust principal component analysis (2009) (preprint)
12. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics 17, 520 (2001)
13. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
14. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 888–905 (2002)
15. Nie, F., Xu, D., Tsang, I., Zhang, C.: Spectral embedded clustering. In: Proceedings of the 21st International Joint Conference on Artifical intelligence, pp. 1181–1186. Morgan Kaufmann Publishers Inc., San Francisco (2009)
16. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Proc. Neural Info. Processing Systems (2003)
17. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: Proc. Int'l Conf. Machine Learning (2003)
18. Efron, B., Hastie, T., Johnstone, L., Tibshirani, R.: Least angle regression. Annals of Statistics 32, 407–499 (2004)

A Novel Stability Based Feature Selection Framework for k-means Clustering*

Dimitrios Mavroeidis and Elena Marchiori

Institute for Computing and Information Sciences,
Radboud University Nijmegen, The Netherlands

Abstract. Stability of a learning algorithm with respect to small input perturbations is an important property, as it implies the derived models to be robust with respect to the presence of noisy features and/or data sample fluctuations. In this paper we explore the effect of stability optimization in the standard feature selection process for the continuous (PCA-based) k-means clustering problem. Interestingly, we derive that stability maximization naturally introduces a tradeoff between cluster separation and variance, leading to the selection of features that have a high cluster separation index that is not artificially inflated by the feature's variance. The proposed algorithmic setup is based on a Sparse PCA approach, that selects the features that maximize stability in a greedy fashion. In our study, we also analyze several properties of Sparse PCA relevant to stability that promote Sparse PCA as a viable feature selection mechanism for clustering. The practical relevance of the proposed method is demonstrated in the context of cancer research, where we consider the problem of detecting potential tumor biomarkers using microarray gene expression data. The application of our method to a leukemia dataset shows that the tradeoff between cluster separation and variance leads to the selection of features corresponding to important biomarker genes. Some of them have relative low variance and are not detected without the direct optimization of stability in Sparse PCA based k-means.

1 Introduction

The stability of a learning algorithm with respect to small input perturbations is generally considered a desired property of learning algorithms, as it ensures that the derived models are robust and are not significantly affected by noisy features or data sample fluctuations. Based on these motivations, the notion of stability has been employed by several popular machine learning paradigms (such as Bagging) and it has been the central theme in several studies that focus both on the theoretical study of stability and the development of practical stability optimizing algorithms. Albeit the considerable amount of research that has been devoted to the study of stability, the interplay between clustering stability and feature selection has not been substantially investigated. This is because most feature selection frameworks do not take into account the contribution of the features to the variance of the derived models and solely evaluate the "relevance" of each feature to the target class structure. This may result in suboptimal models since

* This work was partially supported by the Netherlands Organization for Scientific Research (NWO) within NWO project 612.066.927.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 421–436, 2011.

prediction error is, as illustrated by the bias-variance decomposition, affected by both the relevance of each feature (bias) and its contribution to the stability (variance) of the resulting data model. These considerations, that are also discussed in [13] motivate the study for practical feature selection algorithms that achieve the right balance between the bias-variance tradeoff and optimize the predictive ability of the resulting models.

In the context of this work we undertake this challenge and explore the potentials of performing feature selection with the general purpose of maximizing the stability of the continuous (PCA-based) k-means clustering output[1]. The proposed analysis is performed at a theoretical, algorithmic and empirical level, which are summarized in the sequel. From the theoretical point of view, we demonstrate that stability maximization, naturally leads to a cluster separation vs. feature variance trade off that results in the selection of features that have a high cluster separation index that is not artificially inflated by the feature's variance. This conceptual contribution brings new insights to the theoretical properties of stability, provides practitioners with a clear understanding as to when the stability maximizing objective is appropriate in a specific application context and also allows for the effective interpretation of the success (or possible failure) of the stability based feature selection process.

From the algorithmic point of view, we propose a Sparse PCA formulation for selecting the relevant features that maximize the stability of the continuous clustering solution. Sparse PCA presents a natural choice, since the continuous k-means solution is derived by the principal components, i.e. the dominant eigenvectors of the feature covariance matrix [5]. In our study of Stable Sparse PCA we derive several interesting results that are related to the suitability of Sparse PCA for feature selection in clustering and also, to the stability of the Sparse PCA output. Specifically, we demonstrate that Sparse PCA can be derived as a continuous relaxation to a feature selection problem that optimizes for a cluster separation index. Moreover, we show that double centering the data before the application of Sparse PCA, leads to a "two-way" stability property. I.e. the stability of the instance-clusters becomes equal to the stability of the feature-clusters. This is an important observation that complements our work with data mining algorithms that utilize the feature clusters in the data mining process. Finally, we propose a novel "two-way" stable Sparse PCA algorithm that relies on a greedy lower bound optimization. These results can be considered as side-contributions to our understanding of Sparse PCA as a feature selection mechanism for clustering.

Empirically, we verify the proposed Stable Sparse PCA framework in the context of Cancer Research. In our experiments we have employed four publicly available microarray datasets that are related to the identification of certain cancer types. The experiments demonstrate that the proposed Stable Sparse PCA method is competitive and often superior to state-of-the-art feature selection methods. In particular, we consider the problem of detecting potential tumor biomarkers using microarray gene expression data. Application of our method to leukemia gene expression data shows that the tradeoff between cluster separation and variance leads to the selection of features corresponding to important biomarker genes. Some of them have relative low variance and are not detected without the direct optimization of stability.

[1] With the term continuous k-means clustering problem we refer to the continuous relaxation approach for approximating k-means [5].

2 Spectral k-means

K-means clustering is arguably the most popular clustering algorithm among data mining practitioners, and albeit its introduction more than 50 years ago, it still constitutes an active area of research. The goal of K-means is to find the clustering that minimizes the sum of squared distances of the elements of each cluster to the cluster centers. Formally this objective can be stated as: $J_K = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2$ where we consider x_i to be the instance vectors, μ_k the respective cluster centers and C_k, to denote the clusters.

The most popular heuristic for approximating J_K is the standard Lloyd's algorithm, that starts with a random initial guess of the cluster center and iteratively converges using an EM-style iterative process to a local optima of the k-means objective. In the context of this work, we focus on a different approximation scheme for the k-means objective that is based on the continuous (spectral) relaxation of the discrete cluster assignment vector[5]. The Spectral relaxation allows us to study the stability of the clustering output using the advanced results of matrix perturbation theory [18].

In order to illustrate spectral k-means, we recall from [5] that the k-means problem can be written in equivalent form as: $J_K = \mathbf{Trace}(X_{fc}^T X_{fc}) - \frac{1}{2} J_D$ where X_{fc} is the input $m \times n$ feature-instance matrix, with centered features (rows), and J_D in the 2-cluster case (clusters c_1 and c_2 with sizes n_1 and n_2) is defined as:

$$J_D = \frac{n_1 n_2}{n} \left[2\frac{d(c_1,c_2)}{n_1 n_2} - \frac{d(c_1,c_1)}{n_1^2} - \frac{d(c_2,c_2)}{n_2^2} \right] \quad (1)$$

with $d(c_k,c_l) = \sum_{i \in C_k, j \in C_l} ||x_i - x_j||^2$. Moreover, in [5] it is demonstrated that $J_D = 2\mathbf{Trace}(Q_{K-1}^T X_{fc}^T X_{fc} Q_{K-1})$, where Q_{K-1} is a $n \times (K-1)$ matrix (n =#inst., K =#clust.) that contains the discrete cluster labels (for the discrete cluster values of matrix Q_{K-1} we refer to [5]).

Based on the afore equations, the minimization of J_K is equivalent to the maximization of J_D. By applying the continuous relaxation to J_D, the *continuous* solution is derived by projecting the data to the $k-1$ principal eigenvectors, i.e. the $k-1$ dominant eigenvectors of the Covariance matrix that correspond to the largest eigenvalues. Naturally, the spectral solution will contain the continuous values and an extra step needs to be applied to discretize the continuous cluster assignments with a popular heuristic being the application of standard Lloyd's k-means to the reduced principal eigenspace.

It can be noticed that the minimization of J_K is equivalent to the maximization of J_D because $\mathbf{Trace}(X_{fc}^T X_{fc})$ is a constant that is equal to the (scaled) sum of variances of the available features[2]. In a feature selection setup this term will not remain constant since different features may have different variances, unless the data are appropriately preprocessed such that they have equal variances.

The stability of Spectral k-means can be evaluated using Matrix Perturbation Theory [18]. The relevant theorems designate that the stability of the continuous Spectral k-means solution depends on the size of the eigengap $\lambda_{k-1} - \lambda_k$ between the $k-1$ and the k largest eigenvalues of the relevant matrix with a larger eigengap implying improved stability. Thus, a stability optimizing algorithm should aim to maximize this eigengap.

[2] This is because $\mathbf{Trace}(X_{fc}^T X_{fc}) = \mathbf{Trace}(X_{fc} X_{fc}^T)$.

3 Stable Sparse PCA

3.1 Stability Maximizing Objective and the Cluster Separation/Variance Tradeoff

We will now move on to define the appropriate optimization objective for feature selection that maximizes the stability of the Spectral k-means clustering output. The proposed formulation is based on the Sparse PCA approach in [3], with the appropriate modifications that account for stability maximization.

In order to optimize for stability, we incorporate a term that accounts for the difference between the two largest eigenvalues. Since the aim is to distinguish between the two largest eigenvalues and not between λ_{k-1} and λ_k, our framework initially considers the two-way clustering problem, and extends for $k > 2$-way clustering using a deflation method analytically described in Section 4.3. In this manner, the proposed framework can select a different subset of features at each sequential step (for each eigenvector) thus possibly identifying different feature subsets for separating between different
clusters.

For facilitating the optimization problem we consider the average difference between the largest eigenvalue with the rest. I.e. $\frac{1}{n}\sum_{i=1}^{n}(\lambda_1 - \lambda_i)$ instead of $\lambda_1 - \lambda_2$. Although this formulation will not directly optimize for the difference between the largest eigenvalues, the objective will have a stronger incentive for minimizing the eigenvalues that are closer to λ_1 since they will contribute more to the maximization of the average difference. As we will illustrate in this section, the difference between the largest and the consecutive eigenvalues gives rise to a tradeoff between maximizing the distance between clusters and the feature variances. This balancing essentially imposes a variance based threshold on a cluster separation index and utterly selects the features that optimize the harder separation objective.

Before we move on to define our objective function we will clarify the notation we will use. We will denote X as our input $m \times n$ (feature-instance) matrix, $C_n = (I - e_n e_n^T/n)$, denotes the standard row (feature) centering matrix, u is a vector of length m, with $u(i) = 1$ if feature i is retained in the final solution. $\mathbf{diag}(u)$ is an $m \times m$ diagonal matrix with vector u in its diagonal (i.e. $u(i,i) = 0$ if feature i is removed, otherwise it is equal to 1), and $\mathbf{card}(u)$ is equal to the number of non-zero elements of u (i.e. the number of features that are selected. It can be observed that the multiplication $\mathbf{diag}(u)X$ essentially removes the features that correspond to $u(i) = 0$. Finally we will denote the column (feature)-centered matrix as $X_{fc} = XC_n$ and also $x_{fc}(i)$ as a $n \times 1$ vector that contains the centered vector-representation of feature i (notice we represent $x_{fc}(i)$ as a column vector although it corresponds to row i of matrix X_{fc}).

Based on the afore notation, the covariance matrix after feature selection (omitting term $1/n$ that does not affect our optimization problem) is defined as:

$$Cov = \mathbf{diag}(u)X_{fc}X_{fc}^T\mathbf{diag}(u)$$

We now define the stability maximizing objective as:

$$\begin{aligned} Obj &= \max_{u\in\{0,1\}^m} \left(\tfrac{1}{n}\textstyle\sum_{i=1}^{n}(\lambda_1(Cov)-\lambda_i(Cov))\right) \\ &= \max_{u\in\{0,1\}^m} \left(\tfrac{n-1}{n}\lambda_1(Cov)-\tfrac{1}{n}\textstyle\sum_{i=2}^{n}\lambda_i(Cov))\right) \\ &= \max_{u\in\{0,1\}^m} \left(\lambda_1(Cov)-\tfrac{1}{n}\mathbf{Trace}(Cov)\right) \end{aligned} \tag{2}$$

Based on the ability to express the k-means objective using the clustering separation index J_D (as analyzed in Section 2), we can derive the afore objective as a continuous relaxation to the following feature selection clustering objective:

$$\max_{u\in\{0,1\}^m} \frac{n_1 n_2}{n}\left[2\frac{d^{(u)}(c_1,c_2)}{n_1 n_2}-\frac{d^{(u)}(c_1,c_1)}{n_1^2}-\frac{d^{(u)}(c_2,c_2)}{n_2^2}\right]-\sum_{i=1}^{m} u_i\cdot \mathrm{var}(f_i) \tag{3}$$

where $d^{(u)}(c_i,c_j)=\sum_{k\in c_i}\sum_{l\in c_j}(x_k^{(u)}-x_l^{(u)})^2$ and $x^{(u)}$ denotes the representation of an instance after feature selection (i.e. only the selected features are taken into account when computing the respective distances). Moreover, $\mathrm{var}(f_i)$ denotes the variance of feature i. Notice that the cluster separation index is essentially the J_D of formula 1 after feature selection. The proof of the relationship between the Objectives 2,3 is mostly based on the derivations made within [5] and is omitted due to space limitations.

Based on the afore analysis, we have demonstrated that stability optimization leads to the introduction of a cluster separation vs. variance tradeoff in the feature selection process. In this manner the features that are selected will have high cluster separation value and among features with equal cluster separation value the ones with the smaller variance will be selected. The novelty of the proposed objective resides in the fact that it explicitly penalizes high feature variance and it leads to the selection of the feature subset that has high cluster separation index and low variance. Although this seems to contradict a basic rule of thumb in feature selection that considers features with high variance to be more helpful in separating between clusters (notably, the selection of features with high variance is commonly used as a baseline in the empirical evaluation of feature selection algorithms), our framework can be justified by the view of feature selection as a *variance reduction* process (as done in [13]). In this conceptual approach, feature selection improves the quality of a learning algorithm when it achieves the reduction of variance without significantly increasing the algorithm's bias. Interestingly, under this paradigm the contribution of each feature to the bias-variance of the output model is more important that the exact identification of the relevant/non-relevant features (i.e. a relevant feature that contributes highly to the model's variance may not be desirable).

The Stable Sparse PCA objective formulation currently accounts only for the maximization of the stability of the instance-clustering output. We use the term"solely" as we will demonstrate in the next section that this objective can be extended such that it simultaneously optimizes the stability of both instance and feature clusters.

3.2 Two-Way Stability

Several data mining frameworks employ the clustering of the features as an important component within the general data mining process. One such example is bi-clustering,

or co-clustering [4] where one tries to cluster simultaneously the features and the instances for identifying the clusters of features that can be used for describing the instance clusters. In these application contexts the stability of the clustering of the features is of central importance, since an unstable cluster structure could result in spurious feature clusters that are sensitive to noise or data sample variations.

Based on these motivations, we will present here the necessary extensions that are needed such that the proposed Stable Sparse PCA objective, optimizes concurrently both for the stability of the instance and feature clusters. We will furtheron refer to this type of concurrent stability optimization as "two-way" stability. As we illustrate in the following lemma, two-way stability can be achieved by employing double-centering, a popular data processing technique. Double centering essentially centers both rows and the columns of the data matrix such that they have zero mean. Based on double-centering the stability between the instance-clusters and feature clusters becomes equivalent. This effect is demonstrated in the following lemma whose proof can be found in the appendix.

Lemma 1. *Let X be our input $m \times n$ feature-instance data matrix. If X is double-centered, then the stability of spectral k-means applied on the instances is equivalent to the stability of spectral k-means applied on the features.*

Based on this observation we can extend the Stable Sparse PCA objective such that it optimizes for two-way stability. In order to achieve this goal we will define the double centered covariance matrix (omitting again the $1/n$ factor) as:

$$Cov = C_m^u X_{fc} X_{fc}^T C_m^u \tag{4}$$

The notation is the same as in the previous section, with the addition of C_m^u that is a matrix that performs instance-centering after feature selection, i.e. it is defined as $C_m^u = \mathbf{diag}(u)(I - \frac{1}{\mathbf{card}(u)} e_m e_m^T)\mathbf{diag}(u)$. It can be observed that if we consider the multiplication $C_m^u X$, the instances (columns) of matrix X are centered *after* the removal of features that correspond to $u(i) = 0$. The two-way stability optimizing objective is now defined simply by replacing the new Cov matrix in the optimization problem 2. Having defined the two-way stable Sparse PCA objective, we will move on in the next section for defining the appropriate efficient optimization framework for performing feature selection.

4 Optimization Framework

4.1 Useful Bounds for Optimizing Stability

Sparse PCA problems are known to be computationally hard and several approximation schemes have been developed for tackling them. In the context of this work we adopt the general approach of performing a greedy forward search that optimizes a lower bound of the stability maximizing objective. This general approach has been also adopted by other Sparse PCA algorithms (such as [3]). The derived bound is summarized in Theorem 1. In the theorem statement we use the same notation as in Section 3.1: **card** denotes the cardinality of a set, $x_{fc}(i)$ denotes the centered representation of a feature

and C_m^u is a matrix that performs instance-centering after feature selection, i.e. it is defined as $C_m^u = \mathbf{diag}(u)(I - \frac{1}{\mathbf{card}(u)} e_m e_m^T)\mathbf{diag}(u)$. The bound is derived for the more complex two-way stable objective. Based on the proof, a simpler bound for the one-way stability case can also be obtained.

Theorem 1. *Let I be a set of features and m a feature such that m does not belong to set I. Moreover, let v denote the dominant eigenvector of matrix $X_{fc}^T C_m^u X_{fc}$ as computed using features in set I. Then, the following lower bound can be derived:*

$$Obj(I \cup \{m\}) \geq Obj(I) + B$$

where

$$\begin{aligned} B = (1 - \tfrac{1}{\boldsymbol{card}(I)+1})[(v^T x_{fc}(m)^2 & - \tfrac{1}{n} x_{fc}(m)^T x_{fc}(m)] \\ - \tfrac{2}{\boldsymbol{card}(I)+1}[(\textstyle\sum_{i\in I} v^T x_{fc}(i)) v^T x_{fc}(m) & - \tfrac{1}{n}(\textstyle\sum_{i\in I} x_{fc}(i))^T x_{fc}(m)] \\ + \tfrac{1}{\boldsymbol{card}(I)(\boldsymbol{card}(I)+1)}[(v^T \textstyle\sum_{i\in I} x_{fc}(i))^2 & - \tfrac{1}{n}(\textstyle\sum_{i\in I} x_{fc}(i))^T (\textstyle\sum_{i\in I} x_{fc}(i))] \end{aligned}$$

It can be observed that the computational cost of this bound is dominated by the cost of computing the dominant eigenvector v of matrix $X_{fc}^T C_m^u X_{fc}$. The suitability of this bound for selecting the feature subset that maximizes for two-way stability is illustrated in the experiments section.

4.2 Greedy Solutions

In order to design efficient approximation schemes for the Stable Sparse PCA objective, we turn to greedy approaches. The proposed greedy algorithm is essentially an adaptation of the greedy strategies proposed in [3] that takes into account for the two distinct elements of our framework (double-centering and two-way stability). Our greedy algorithm also takes advantage of the lower bound derived in Theorem 1 and performs the greedy search *without explicitly computing the objective function* for each candidate feature.

The complexity of Algorithm 1 is $O(np^3 + n^2p^2 + nm^2)$ where p is the number of selected features and n the number of instances. This is because, at each step l (when selecting the l^{th} feature) in order to compute the bound we must double center the data matrix $O(nl^2 + n^2 l)$ (complexity of double centering an $n \times l$ matrix) and then compute the maximum eigenvalue of a matrix of size $n \times n$ which is $O(n^2)$ only once per greedy step. The candidate feature is selected based on the maximum angle between certain vector-pairs of sizes $n \times 1$ that induce a computational cost of $O(n(m-l))$. Since, double centering and the maximum eigenvalue is computed only once per greedy step of the algorithm, the total complexity will be $O(np^3 + n^2p^2 + nm^2)$.

It should be noted that because of double centering, the proposed algorithm is not able to select the initial feature (all features would appear to have quality equal to zero), thus the greedy algorithm is initialized with the feature that maximizes O_1 component of the objective (as defined within the proof of Theorem 1). It can be easily observed that this will essentially be the feature that has maximum variance. The algorithm terminates when the desired number of features p is selected.

Algorithm 1. (X,p)

1: Initialize with index I_{k_0} where $i_0 = argmax_{j \in I} O_1\{j\}$.
2: **repeat**
3: Compute $i_k = argmax_{i \in I_k^c} B(i,I_k)$.($B(i,I_k)$ is the lower bound of theorem 1)
4: Set $I_{k+1} = I_k \cup \{i_k\}$.
5: **until card**$(I_{k+1}) = p$.

4.3 Efficient Deflation for Multiple Clusters

In order to extend our framework for multiple clusters ($k > 2$), we consider the use of deflation. Although deflation is a rather straight forward approach for extracting multiple eigenvectors in the full feature case, it presents certain challenges in the context of sparse methods. These challenges are analytically illustrated in [11] where several deflation methods and their properties were thoroughly analyzed. Based on [11], one could simply employ one of the proposed methods, such as the Schur complement deflation, for computing the sequential sparse eigenvectors of the Covariance matrix. One issue with employing an "of-the-shelve" approach is that we would need to compute the Cholesky decomposition of the covariance matrix, in order to derive the new centered feature representations that are consequently employed in the bound computations (i.e. the $x_{fc}(i)$ in Theorem 1). This would affect the computational cost of the proposed method as it would include a $O(m^3)$ term for the Cholesky decomposition. In order to avoid this computational cost we propose a deflation process that is directly applied on the centered feature matrix X_{fc} using the dominant eigenvector of matrix $X_{fc}^T C_m^u X_{fc}$. As we will demonstrate, the proposed deflation is essentially equivalent to Schur complement deflation.

$$X_{fc}^{(t)} = X_{fc}^{(t-1)}(I - v_t v_t^T) \tag{5}$$

Starting from $t = 0$, the original input matrix of centered features, $X_{cf}^{(0)}$ is used for computing $Cov^{(0)}$ and for deriving the subset of features (as encoded in $u(t=0) \in \{0,1\}^m$) that optimizes the Stable Sparse PCA objective. Based on the selected features, the dominant eigenvector v_1 of $(X_{fc}^{(0)})^T C_m^{u(0)} X_{fc}^{(0)}$ is derived. Consecutively, we can employ the deflation formula for computing all the necessary eigenvectors. An interesting property of the deflation process is that at each sequential step, a different feature subset may be derived, thus giving the flexibility to the feature selection algorithm to select different feature subsets for separating between different clusters.

In order to illustrate the appropriateness of the afore proposed deflation method, we demonstrate that it is equivalent to a Schur complement deflation. The proof of this theorem can be found in the appendix.

Theorem 2. *The deflation procedure defined in equation 5 is equivalent to a Schur complement deflation on the feature covariance matrix.*

5 Related Work

The proposed framework is conceptually related to the work [13] that attributes the success of feature selection methods to the reduction of the data model variance. Under this

approach, features should not be selected simply by assessing their relevance to the target class but by considering the contribution that the features have to the bias-variance tradeoff of the learned model. That is, weakly relevant features that contribute much to the variance of the model should be excluded, while borderline-relevant features with low variance contribution can present good candidates for inclusion. In our study we adopt this principle and derive a criterion that selects features based on their contribution to cluster separation, weighted against their variance. Interestingly the cluster-separation variance tradeoff is derived through a stability maximizing objective.

In the relevant data mining literature, the term "stability of feature selection" is employed in a different manner and commonly refers to the robustness of the feature process itself, i.e. the ability of a feature selection algorithm to select the same feature with respect to noise, or data sample variations. The intuitiveness of this requirement has resulted in several works that study the stability of feature selection algorithms [9,7,15,10,21]. Our work is substantially different from these approaches, since it focuses on the effect of feature selection to the stability of the clustering output and not the feature selection process itself. Albeit this important differentiation, the "two-way" stability optimization framework can be employed in conjunction with the works [21,10]. This is because these methods employ the clustering structure of the features and perform feature selection at the clustering level (i.e. they select the relevant/stable feature clusters). In this context our approach can be employed as a preprocessing step that stabilizes the feature clusters thus enhancing the robustness of these methods.

The idea of selecting the features that optimize for an eigengap of a certain input matrix was also put forward in [20]. In this work the authors propose a continuous feature weighting scheme that achieves sparsity in an indirect manner. As opposed to [20] we conduct a detailed analysis on the properties of a stability maximizing objective in the context of k-means clustering. Moreover, we explicitly formulate our algorithm as a discrete feature selection and propose a novel Sparse PCA approach for selecting the appropriate features.

In [12] the authors optimize for stability by removing the features that contribute maximally to the variance of the derived model. This approach does not take into account the relevance of each feature and thus high quality features may be removed. In contrast, our approach explicitly takes into account the cluster separation quality of each feature that is weighted against the feature-variance.

In order to empirically validate our approach we compare our approach to the popular Laplacian score [8] and the recently proposed MCFS algorithm [1]. Although these algorithms are not conceptually relevant to the proposed framework, they provide a good basis for demonstrating that the proposed feature selection framework can achieve comparable performance with state-of-the-art algorithms.

6 Experiments

In the context of the Cancer research application, we have experimented with four publicly available microarray datasets that were obtained from `http://algorithmics.molgen.mpg.de/Static/Supplements/CompCancer/datasets.htm`. The employed datasets are summarized in the following table:

Name	Description	#Instances	#Features	#Classes
Chen-2002	Liver Cancer	179	85	2
Golub-1999-v2	Leukemia	72	1877	3
Pomeroy-2002-v2	Central Nervous System Tumors	42	1379	5
Ramaswamy-2001	Multiple Cancer Types	190	1363	14

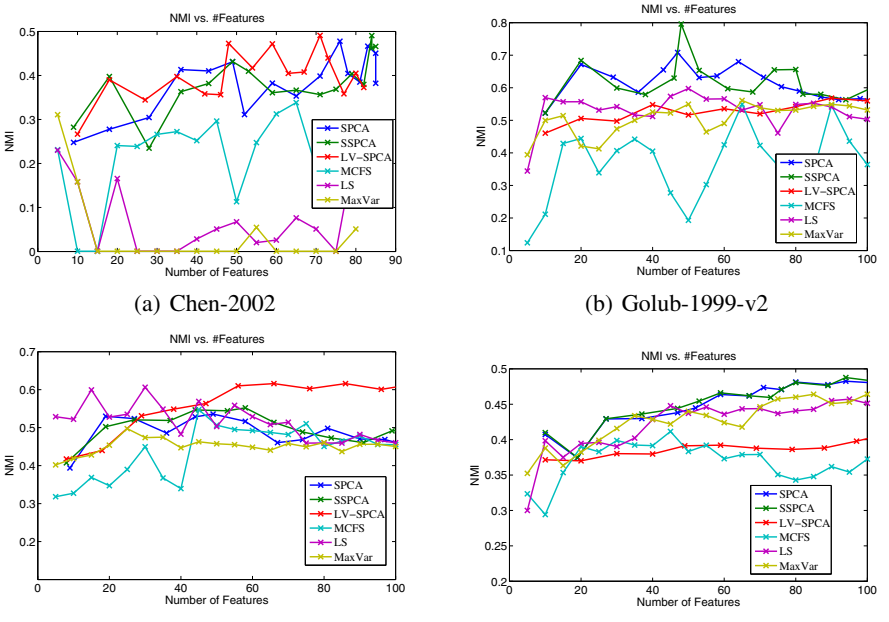
We compare our feature selection framework against *Laplacian Score* [8], the recently proposed *MCFS* algorithm [1] and the simple heuristic of selecting the features that have maximal variance. In order to conduct the comparison, we employ the selected feature subsets in the context of a k-means clustering algorithm and compute the achieved cluster quality using Normalized Mutual Information *(NMI)*.

It should be noted that our method differs substantially from the *Laplacian Score* and *MCFS*. The main difference is that our method is "faithful" to the k-means objective, while the methods we compare against construct a Laplacian matrix for measuring the relevance of the features. It is evident that the construction of the Laplacian matrix can substantially influence the results (i.e. by considering different similarity functions, Gaussian vs. simple inner-product or different types of graphs, k-nn Graphs vs. Full Graphs). Thus, a major factor that determines which method is more suitable for different application contexts depends on the ability to construct/tune an appropriate Laplacian matrix and also on the properties of the underlying clustering structure of the data.

For constructing the Laplacian matrix for the *MCFS* and *Laplacian Score* we employ the cosine similarity for computing the instance-similarity matrix W and then construct a k-nearest neighbor graph with $k = 5$. These settings are similar to the ones recommended in [1]. Moreover, the number of Laplacian eigenvectors that were employed within *MCFS* was set to be equal to the number of clusters.

Our method constructs the continuous cluster indicators using 2 sparse eigenvectors in all datasets (instead of #Cluster-1). In three out of four dataset this is different than the (#Cluster-1) that is recommended by the "pure" Spectral k-means approach. In two cases (Pomeroy and Ramaswamy datasets) we have employed only two sparse eigenvectors in order to obtain comparable results for a small number of features. This is because the proposed framework can select different features for each eigenvector, thus even when selecting a small number of features for each sparse eigenvector, the total number of features can be much larger. In the Chen-2002 dataset we employed 2 sparse eigenvectors for performance reasons, since we observed that using only 1 eigenvector did not suffice to obtain a good clustering performance. We should finally note that the standard Lloyd's k-means algorithms was used for discretizing the continuous results and also that we have employed the "two-way" stable version of our objective.

In the experiments we have also explored the possibilities of introducing different tradeoffs between cluster-separation and feature variance. More precisely we have experimented with three objectives: The standard cluster-separation vs. variance tradeoff that is derived by maximizing the average eigengap between the largest and the consecutive eigenvalues. $\lambda_1(Cov) - \frac{1}{n}\mathbf{Trace}(Cov)$ (denoted in the Figures as *SSPCA*), a "pure" Sparse PCA approach that maximizes solely $\lambda_1(Cov)$ (denoted in the Figures as *SPCA*), and a "low variance" approach that penalizes heavily variance using $\lambda_1(Cov) - \gamma\mathbf{Trace}(Cov)$, where $\gamma = \frac{\lambda_1(Cov_{full})}{\mathbf{Trace}(Cov_{full})}$ i.e. it is equal to the ratio of the



(a) Chen-2002 (b) Golub-1999-v2

(c) Pomeroy-2002-v2 (d) Ramaswamy-2001

Fig. 1. Comparative Study of Clustering Quality

maximum eigenvalue to the Trace of the original full-feature Covariance matrix, before feature selection (denoted in the Figures as *LV-SPCA*).

In Figure 1 we can observe that the clustering quality is competitive and often superior against the relevant feature selection methods. More precisely, in all Figures at least one of the three Sparse PCA methods is superior with the exception of Figure 1(c), where the Laplacian Score is better for small feature sizes. Moreover, we can observe a mixed behavior with respect to the appropriate level of variance penalization, with *LV-SPCA* demonstrating a very good performance in two out of four experiments.

Apart from the indirect evaluation of the accuracy of feature selection (through clustering quality) we also investigate whether the selected features can provide insights into the underlying problem under study. For this purpose we focus on the Golub-1999-v2 dataset [6] that is related to Acute lymphoblastic leukemia (ALL), which is the most common pediatric cancer, accounting for 30% of all pediatric malignancies.

Golub's dataset [6] consists of bone marrow sample from acute leukemia patients, involving myeloid leukemia (AML) and two sub-types of acute lymphoblastic leukemia (ALL), B-cell and T-cell ALL. The analyzed Golub-1999-v2 dataset consists of 38 ALL-B, 9 ALL-T and 25 AML samples and 1877 genes.

We compared the top 5 features selected in the first and second eigenvector generated using two different versions of the proposed algorithm *SPCA* (where solely $\lambda_1(Cov)$ is optimized with no variance correction) and *LV-SPCA* that employs a strong variance correction threshold. Table 1 shows the list of genes. As perhaps one could expect, the variance of the genes only selected using *LV-SPCA* is in general smaller than the one of

those selected using *SPCA*. In order to investigate the relevance of the selected genes for the disease under study, we performed a literature research on the three genes uniquely identified by the proposed method with direct optimization of stability.

Genes Uniquely Selected by LV-SPCA. Gene number 458 has ID M21005_at and corresponds to the S100 calcium binding protein A8 (calgranulin A). A very recent experimental investigation has been performed in [14] which suggests that the expression of S100A8 in leukemic cells is a predictor of low survival. This gene was not found among top 100 of *MCFS*, and was ranked by *Laplacian Score* as 66th. Furthermore, this gene was also not predicted as relevant by other gene ranking methods (see *http://genomics10.bu.edu/yangsu/rankgene/compare-ALL-AML-all-top100.html#ranks_table*).

Gene number 1614 has ID Y00787_s_at and corresponds to the Interleukin-8 precursor. In [16] it has been suggested that Interleukin-8 upregulation may play a role in the pathogenesis of T-cell acute lymphoblastic leukemia.

Gene number 1613 has ID M28130_rna1_s_at and corresponds to the Interleukin 8 (IL8) gene. It has been suggested that IL-8 may function as a significant regulatory factor within the tumor microenvironment. Recently, IL-8 signaling has been implicated in regulating the transcriptional activity of the androgen receptor, underpinning the transition to an androgen-independent proliferation of prostate cancer cells. In addition, stress and drug-induced IL-8 signaling has been shown to confer chemotherapeutic resistance in cancer cells. Therefore, inhibiting the effects of IL-8 signaling may be a significant therapeutic intervention in targeting the tumor microenvironment [19]. Indeed, Interleukin 8 (IL-8) is currently being applied in various subspecialties of medicine either for the purpose of rapid diagnosis or as a predictor of prognosis: in [17] an overview of current evidence is provided suggesting that Interleukin 8 (IL-8) may serve as a useful biomarker.

Genes Uniquely Selected by SPCA. Gene number 607 has ID M91036_rna1_at and corresponds to the G-gamma globin gene. Gene number 1798 has ID U01317_cds4_at and corresponds to the Delta-globin gene. We did not find strong evidence of a relation of these two genes with the leukemia pathogenesis and pharmacology.

Gene number 493 has ID U10685_at and corresponds to the MAGE A10 gene. The mammalian members of the MAGE (melanoma-associated antigen) gene family were originally described as completely silent in normal adult tissues, with the exception of male germ cells and, for some of them, placenta. By contrast, these genes were expressed in various kinds of tumors. However, other members of the family were recently

Table 1. Index of top 5 genes selected by the method in the first and second eigenvector for *SPCA* and *LV-SPCA*. The number between brackets indicates the position of the gene in the list of gene sorted in decreasing with respect to the variance.

Method	Eigenvector	Feat1	Feat2	Feat3	Feat4	Feat5
SPCA	i=1	1623 (1)	1194 (5)	**493** (2)	1106 (17)	672 (33)
SPCA	i=2	**607** (8)	1734 (9)	435 (15)	**1798** (16)	1756 (27)
LV-SPCA	i=1	1623 (1)	1194 (5)	**458** (18)	672 (33)	1106 (17)
LV-SPCA	i=2	**1614** (4)	1734 (9)	**1613** (23)	435 (15)	1756 (27)

found to be expressed in normal cells, indicating that the family is larger and more disparate than initially expected [2].

The above observations indicate the effectiveness of optimizing stability for unsupervised feature selection with respect to the detection of features strongly related to the pathogenesis and pharmacology of the disease under study.

7 Conclusions and Further Work

In conclusion, we have proposed a novel feature selection framework that maximizes the stability of Spectral k-means. The semantics of the proposed framework are analyzed in detail and it is demonstrated that stability maximization naturally leads to a cluster-separation vs. variance tradeoff. As a matter of further work, we aim in extending our framework to Kernel k-means and also to Spectral Clustering algorithms that employ the Laplacian matrix.

Appendix

Proof (Proof of Lemma 1). Based on [5], the continuous solution for the instance clusters is derived by the $k-1$ dominant eigenvectors of matrix $X_{fc}^T X_{fc}$, where X_{fc} is a feature-instance matrix with the rows (features) being centered. Since X is double-centered the sum of rows and columns of X will be equal to 0, i.e. $\sum_i X_{ij} = \sum_j X_{ij} = 0$. Thus, the continuous solution of Spectral k-means (for instance clustering) will be derived by the $k-1$ dominant eigenvectors of matrix $X^T X$.

Analogously, the continuous solution for the feature clusters is derived by the $k-1$ dominant eigenvectors of matrix $X_{ic} X_{ic}^T$, where X_{ic} is a feature-instance matrix with the columns (instances) being centered. Since X is double-centered, we will have that the instance-centered matrix X_{ic} will be equal to X, i.e. $X_{ic} = X$. Thus, the continuous cluster solution will be derived by the dominant eigenvectors of matrix XX^T.

Using basic linear algebra one can easily derive that the matrices XX^T and $X^T X$ have exactly the same eigenvalues.

Thus $\lambda_{k-1}(XX^T) - \lambda_k(XX^T) = \lambda_{k-1}(X^T X) - \lambda_k(X^T X)$, and the stability of the relevant eigenspaces will be equivalent.

Proof (Proof of Theorem 1). We will start by decomposing the components $\lambda_1(Cov)$ and $\mathbf{Trace}(Cov)$.

For $\lambda_1(Cov)$ we have:

$$\begin{aligned}
\lambda_1(Cov) &= \lambda_1(C_m^u X_{fc} X_{fc}^T C_m^u) \\
&= \lambda_1(X_{fc}^T C_m^u X_{fc}) \\
&= \lambda_1(X_{fc}^T \mathbf{diag}(u)(I - \tfrac{1}{\mathbf{card}(u)} e_m e_m^T)\mathbf{diag}(u) X_{fc}) \\
&= \max_{||v||=1} v^T (X_{fc}^T \mathbf{diag}(u)(I - \tfrac{1}{\mathbf{card}(u)} e_m e_m^T)\mathbf{diag}(u) X_{fc}) v \\
&= \max_{||v||=1} v^T (X_{fc}^T \mathbf{diag}(u) X_{fc}) v \\
&\qquad - \tfrac{1}{\mathbf{card}(u)} v^T (X_{fc}^T \mathbf{diag}(u) e_m e_m^T \mathbf{diag}(u) X_{fc}) v) \\
&= \max_{||v||=1} \sum_{i=1}^{m} u_i (v^T x_{fc}(i))^2 - \tfrac{1}{\mathbf{card}(u)} (v^T \sum_{i=1}^{m} u_i x_{fc}(i))^2
\end{aligned}$$

In the above derivations we have used the following, easily verifiable facts: $\lambda_1(AA^T) = \lambda_1(A^TA)$, $C_m^u = (C_m^u)^T$, $C_m^u = (C_m^u)^2$ and $\mathbf{diag}(u) = (\mathbf{diag}(u))^2$.

For $\mathbf{Tr}(Cov)$ we have:

$$\begin{aligned}\mathbf{Tr}(Cov) &= \mathbf{Tr}(C_m^u X_{fc} X_{fc}^T C_m^u) \\ &= \mathbf{Tr}(X_{fc}^T C_m^u X_{fc}) \\ &= \mathbf{Tr}(X_{fc}\mathbf{diag}(u)(I - \tfrac{1}{\mathbf{card}(u)} e_m e_m^T)\mathbf{diag}(u)X_{fc}) \\ &= \mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)X_{fc}) \\ &\quad - \tfrac{1}{\mathbf{card}(u)}\mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)e_m e_m^T\mathbf{diag}(u)X_{fc})\end{aligned}$$

In these derivations we have used the following properties of the matrix Trace: $\mathbf{Tr}(AA^T) = \mathbf{Tr}(A^TA)$, $\mathbf{Tr}(A+B) = \mathbf{Tr}(A) + \mathbf{Tr}(B)$ and $\mathbf{Tr}(\beta A) = \beta\mathbf{Tr}(A)$.

Now for $\mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)X_{fc})$ we have:

$$\begin{aligned}\mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)X_{fc}) &= \mathbf{Tr}(\mathbf{diag}(u)X_{fc}X_{fc}^T\mathbf{diag}(u)) \\ &= \sum_{i=1}^{m} u_i x_{fc}(i)^T x_{fc}(i)\end{aligned}$$

Finally, for $\frac{1}{\mathbf{card}(u)}\mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)e_m e_m^T\mathbf{diag}(u)X_{fc})$ we have:

$$\begin{aligned}&\tfrac{1}{\mathbf{card}(u)}\mathbf{Tr}(X_{fc}^T\mathbf{diag}(u)e_m e_m^T\mathbf{diag}(u)X_{fc}) = \\ &\tfrac{1}{\mathbf{card}(u)}\mathbf{Tr}(e_m^T\mathbf{diag}(u)X_{fc}X_{fc}^T\mathbf{diag}(u)e_m) = \\ &\tfrac{1}{\mathbf{card}(u)}(\sum_{i=1}^{n} u_i x_{fc}(i))^T(\sum_{i=1}^{m} u_i x_{fc}(i))\end{aligned}$$

Based on the afore derivations we can write the objective function as:

$$\max_{||v||=1}\max_{u\in\{0,1\}^n}[O_1 - \frac{1}{\mathbf{card}(u)}O_2] \tag{6}$$

Where

$$O_1 = \sum_{i=1}^{m} u_i[(v^T x_{fc}(i))^2 - \frac{1}{n}x_{fc}(i)^T x_{fc}(i)]$$

$$O_2 = (v^T\sum_{i=1}^{m} u_i x_{fc}(i))^2 - \frac{1}{n}(\sum_{i=1}^{m} u_i x_{fc}(i))^T(\sum_{i=1}^{m} u_i x_{fc}(i))$$

Based on the derivation of the objective function using O_1 and O_2 and also the fact that for all v such that $||v|| = 1$ it holds that $\lambda_1(A) \geq x^TAx$, the lower bound can be derived.

Proof (Proof of Theorem 2). Recall that in Schur complement deflation, the deflation step is performed as follows:

$$A_t = A_{t-1} - \frac{A_{t-1}x_t x_t^T A_{t-1}}{x_t^T A_{t-1} x_t}$$

Now if we consider that $A_t = X_{fc}^{(t)}(X_{fc}^{(t)})^T$ and also that x_t is the dominant eigenvector of matrix $Cov^{(t-1)}$, we can write:

$$A_t = A_{t-1} - \frac{A_{t-1}x_t x_t^T A_{t-1}}{x_t^T A_{t-1} x_t} \Rightarrow$$

$$X_{fc}^{(t)}(X_{fc}^{(t)})^T = X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T - \frac{X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T x_t x_t^T X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T}{x_t^T X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T x_t} \tag{7}$$

Recall that x_t is the dominant eigenvector of $Cov^{(t-1)}$ that can be written as $Cov^{(t-1)} = C_m^{u(t-1)} X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T C_m^{u(t-1)}$ (i.e. it is based on the selected feature subset $u(t-1)$).

Since $Cov^{(t-1)}$ is a double-centered matrix (its rows and columns are centered through the multiplication with $C_m^{u(t-1)}$), its dominant eigenvector will also be centered, i.e. $(C_m^{u(t-1)})x_t = x_t$. Based on this property, we can write:

$$\begin{aligned} x_t^T X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T x_t &= x_t^T C_m^{u(t-1)} X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T C_m^{u(t-1)} x_t \\ &= x_t^T Cov^{(t-1)} x_t = \lambda_{max}^{(t-1)} \end{aligned} \tag{8}$$

Now, $(X_{fc}^{(t-1)})^T x_t$ can be written as:

$$(X_{fc}^{(t-1)})^T x_t = (X_{fc}^{(t-1)})^T C_m^{u(t-1)} x_t = \sqrt{\lambda_{max}^{(t-1)}} v_t, \tag{9}$$

where v_t is the dominant eigenvector of $(X_{fc}^{(t-1)})^T C_m^{u(t-1)} X_{fc}^{(t-1)}$ and $\lambda_{max}^{(t-1)}$ is the dominant eigenvector of $Cov^{(t-1)}$.

Based on equations 7, 8, 9 we can write

$A_t = A_{t-1} - \frac{A_{t-1}x_t x_t^T A_{t-1}}{x_t^T A_{t-1} x_t} \Rightarrow$

$X_{fc}^{(t)}(X_{fc}^{(t)})^T = X_{fc}^{(t-1)}(X_{fc}^{(t-1)})^T - X_{fc}^{(t-1)} v_t v_t^T (X_{fc}^{(t-1)})^T \Rightarrow$

$X_{fc}^{(t)} = X_{fc}^{(t-1)}(I - v_t v_t^T)$

References

1. Cai, D., Zhang, C., He, X.: Unsupervised feature selection for multi-cluster data. In: ACM SIGKDD (2010)
2. Chomez, P., Backer, O.D., Bertrand, M., Plaen, E.D., Boon, T., Lucas, S.: An overview of the mage gene family with the identification of all human members of the family. Cancer Research 15, 6 (2001)
3. d'Aspremont, A., Bach, F.R., Ghaoui, L.E.: Full regularization path for sparse principal component analysis. In: ICML (2007)
4. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: ACM SIGKDD (2001)
5. Ding, C.H.Q., He, X.: K-means clustering via principal component analysis. In: ICML (2004)
6. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439), 531–537 (1999)
7. Han, Y., Yu, L.: A variance reduction framework for stable feature selection. In: IEEE ICDM (2010)

8. He, X., Cai, D., Niyogi, P.: Laplacian score for feature selection. In: NIPS (2005)
9. Kalousis, A., Prados, J., Hilario, M.: Stability of feature selection algorithms: a study on high-dimensional spaces. Knowl. Inf. Syst. 12(1), 95–116 (2007)
10. Loscalzo, S., Yu, L., Ding, C.H.Q.: Consensus group stable feature selection. In: ACM SIGKDD (2009)
11. Mackey, L.: Deflation methods for sparse pca. In: NIPS (2008)
12. Mavroeidis, D., Vazirgiannis, M.: Stability based sparse LSI/PCA: Incorporating feature selection in LSI and PCA. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 226–237. Springer, Heidelberg (2007)
13. Munson, M.A., Caruana, R.: On feature selection, bias-variance, and bagging. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 144–159. Springer, Heidelberg (2009)
14. Nicolas, E., Ramus, C., Berthier, S., Arlotto, M., Bouamrani, A., Lefebvre, C., Morel, F., Garin, J., Ifrah, N., Berger, F., Cahn, J.Y., Mossuz, P.: Expression of s100a8 in leukemic cells predicts poor survival in de novo aml patients. Leukemia 25, 57–65 (2011)
15. Saeys, Y., Abeel, T., Van de Peer, Y.: Robust feature selection using ensemble feature selection techniques. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 313–325. Springer, Heidelberg (2008)
16. Scupoli, M., Donadelli, M., Cioffi, F., Rossi, M., Perbellini, O., Malpeli, G., Corbioli, S., Vinante, F., Krampera, M., Palmieri, M., Scarpa, A., Ariola, C., Foa, R., Pizzolo, G.: Bone marrow stromal cells and the upregulation of interleukin-8 production in human t-cell acute lymphoblastic leukemia through the cxcl12/cxcr4 axis and the nf-kappab and jnk/ap-1 pathways. Haematologica 93(4), 524–532 (2008)
17. Shahzad, A., Knapp, M., Lang, I., Kohler, G.: Interleukin 8 (il-8) - a universal biomarker? International Archives of Medicine 3(11) (2010)
18. Stewart, G.W., Sun, J.G.: Matrix Perturbation Theory (Computer Science and Scientific Computing). Academic Press, London (1990)
19. Waugh, D., Wilson, C.: The interleukin-8 pathway in cancer. Clinical Cancer Research (2008)
20. Wolf, L., Shashua, A.: Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach. J. Mach. Learn. Res. (2005)
21. Yu, L., Ding, C.H.Q., Loscalzo, S.: Stable feature selection via dense feature groups. In: ACM SIGKDD (2008)

Link Prediction via Matrix Factorization

Aditya Krishna Menon and Charles Elkan

University of California, San Diego
La Jolla, CA 92093
{akmenon,elkan}@cs.ucsd.edu

Abstract. We propose to solve the link prediction problem in graphs using a supervised matrix factorization approach. The model learns latent features from the topological structure of a (possibly directed) graph, and is shown to make better predictions than popular unsupervised scores. We show how these latent features may be combined with optional explicit features for nodes or edges, which yields better performance than using either type of feature exclusively. Finally, we propose a novel approach to address the class imbalance problem which is common in link prediction by directly optimizing for a ranking loss. Our model is optimized with stochastic gradient descent and scales to large graphs. Results on several datasets show the efficacy of our approach.

Keywords: Link prediction, matrix factorization, side information, ranking loss.

1 The Link Prediction Problem

Link prediction is the problem of predicting the presence or absence of edges between nodes of a graph. There are two types of link prediction: (i) *structural*, where the input is a partially observed graph, and we wish to predict the status of edges for unobserved pairs of nodes, and (ii) *temporal*, where we have a sequence of fully observed graphs at various time steps as input, and our goal is to predict the graph state at the next time step. Both problems have important practical applications, such as predicting interactions between pairs of proteins and recommending friends in social networks respectively. This document will focus on the structural link prediction problem, and henceforth, we will use the term "link prediction" to refer to the structural version of the problem.

Link prediction is closely related to the problem of collaborative filtering, where the input is a partially observed matrix of (user, item) preference scores, and the goal is to recommend new items to a user. Collaborative filtering can be seen as a bipartite weighted link prediction problem, where users and items are represented by nodes, and edges between nodes are weighted according to the preference score. More generally, both problems are instances of *dyadic prediction*, which is the problem of predicting a label for the interaction between pairs of entities (or *dyads*) [17]. Despite this connection, there has been limited interaction between the link prediction and collaborative filtering literatures, aside from a few papers that propose to solve one problem using models from the other [7,23].

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 437–452, 2011.

1.1 Challenges in Link Prediction

We point out three challenges in link prediction that a model should ideally address. First, in addition to the topological information of the graph, we sometimes have extra *side information* or covariates for the nodes. For example, in a graph of interaction between pairs of proteins, we might have features describing the biological properties of each protein. This information can be useful in predicting links, especially when a node is only sparsely connected. Since the graph topology and the side information potentially encode different types of information, combining them is expected to give the best performance. However, it is not obvious how best to do this in general. Further, to be flexible, we would like to make the side information only an *optional* component of a model.

Second, link prediction datasets are characterized by extreme *imbalance*: the number of edges known to be present is often significantly less than the number of edges known to be absent. This issue has motivated the use of area under the ROC curve (AUC) as the *de facto* performance measure for link prediction tasks, as, unlike standard 0-1 accuracy, AUC is not influenced by the distribution of the classes. However, the class imbalance still hampers the effectiveness of many models that would otherwise be suitable on balanced data.

Third, it is imperative that models be computationally efficient if they are to scale to graphs with a large number of nodes and/or edges. Such large-scale graphs are characteristic of many real-world applications of link prediction, such as the aforementioned friend-recommendation in social networks.

1.2 Our Contributions

This paper studies the effectiveness of matrix factorization techniques for the structural link prediction problem, inspired by their success in collaborative filtering [20]. We first make a case for matrix factorization to serve as a foundation for a general purpose link prediction model. We explain how it may be thought of as learning *latent features* from the data, and why it can be expected to be more predictive than popular unsupervised scores. We show how latent features may be combined with explicit features for nodes and edges, and in particular how the factorization can be combined with the outputs of other models. Further, we propose a novel mechanism to allow factorization models to overcome the imbalance problem, based on the idea of optimizing for a *ranking loss*. Experimentally, we first show that factorization significantly outperforms several popular unsupervised scores. Next, we demonstrate that while explicit features are usually able to provide better estimates of linking behaviour than implicit features, combining the two can further improve performance. Finally, we show that optimizing for a ranking loss can improve AUC by up to 10%.

Before proceeding, we define the link prediction problem more formally and fix the notation used in this paper.

1.3 Problem Definition and Notation

Formally, structural link prediction has as input a partially observed graph $G \in \{0, 1, ?\}^{n \times n}$, where 0 denotes a *known absent* link, 1 denotes a *known present*

link, and ? denotes an unknown status link. Our goal is to make predictions for the ? entries. The set of observed dyads is denoted by $\mathcal{O} = \{(i,j) : G_{ij} \neq ?\}$, and we use $\mathcal{O}_i$ to mean the observed dyads involving the ith node. In some cases, we may have additional features (or covariates) for dyads and/or individual nodes. We call such features *side information*.

We will use capital variables (e.g. X) to denote matrices and lower-case variables (e.g. x) to denote vectors. We will use x_i to mean the ith row of the matrix X. The Frobenius norm of the matrix X is denoted by $||X||_F^2 = \sum_i ||x_i||_2^2$.

2 Existing Link Prediction Models

At a high level, existing link prediction models fall into two classes: unsupervised and supervised. Unsupervised models compute scores for pairs of nodes based on topological properties of the graph. For example, one such score is the number of common neighbours that two nodes share. Other popular scores are the Adamic-Adar [1] and Katz score [22]. These models use predefined scores that are invariant to the specific structure of the input graph, and thus do not involve any learning. Supervised models, on the other hand, attempt to be directly predictive of link behaviour by learning a parameter vector θ via

$$\min_{\theta} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \ell(G_{ij}, \hat{G}_{ij}(\theta)) + \Omega(\theta), \tag{1}$$

where $\hat{G}_{ij}(\theta)$ is the model's predicted score for the dyad (i,j), $\ell(\cdot,\cdot)$ is a *loss function*, and $\Omega(\cdot)$ is a *regularization* term that prevents overfitting. The choice of these terms depends on the type of model. We list some popular approaches:

1. **Feature-based models.** Suppose each node i in the graph has an associated feature vector $x_i \in \mathbb{R}^d$. Suppose further that each dyad (i,j) has a feature vector $z_{ij} \in \mathbb{R}^D$. Then, we may instantiate Equation 1 via

$$\hat{G}_{ij}(w,v) = L(f_D(z_{ij}; w) + f_M(x_i, x_j; v)) \tag{2}$$

   for appropriate $f_D(\cdot), f_M(\cdot,\cdot)$ acting on dyadic and monadic features respectively, and a link function $L(\cdot)$. Both linear [33] and nonlinear [14,5] choices of $f_D(\cdot), f_M(\cdot,\cdot)$ have been considered. In the linear case, it is standard to let $f_D(z_{ij}; w) = w^T z_{ij}$ and $f_M(x_i, x_j; v) = (v^{(1)})^T x_i + (v^{(2)})^T x_j$, where $v^{(1)} = v^{(2)}$ iff the graph is undirected. The loss is typically either square- or log-loss, and the regularizer typically $\frac{\lambda_w}{2}||w||_2^2 + \frac{\lambda_v}{2}||v||_2^2$. Note also that we can compute multiple unsupervised scores between pairs of nodes (i,j), and treat these as comprising a feature vector z_{ij}.
2. **Graph regularization models.** Here, we assume the existence of node features $x_i \in \mathbb{R}^d$, based on which we construct a kernel $K_{ii'jj'}$ that compares the node pairs (i,j) and (i',j'). We compute the predicted adjacency matrix $\hat{G}$ by constraining that values in this matrix should vary smoothly according

to K. Thus K acts as a graph regularizer, a popular approach in semi-supervised learning [39]. In the framework of Equation 1, we have

$$\Omega(\hat{G}) = \frac{\lambda}{2} \sum_{i,i',j,j'} K_{ii'jj'} (\hat{G}_{ij} - \hat{G}_{i'j'})^2 + \frac{\mu}{2} \sum_{(i,j) \notin \mathcal{O}} \hat{G}_{ij}^2$$

The above is called *link propagation* [19,26]. The performance of such methods depends on the choice of kernel K, which is pre-specified and not learned from the data.

3. **Latent class models.** These models assign each node of the graph to a class, and use the classes to predict the link structure. [4] assumes that nodes interact solely through their class memberships. It is possible to extend this to allow nodes to have membership in multiple classes [3]. These models are largely Bayesian, and so are not directly expressible in the empirical loss framework of Equation 1. Nonetheless, they do learn a matrix of probabilities $P \in \{0,1\}^{C \times C}$, where C is the number of classes, and this is done by placing appropriate priors on P, which may be viewed as a form of regularization.
4. **Latent feature models.** Here, we treat link prediction as a matrix completion problem, and factorize $G \approx L(U \Lambda U^T)$ for some $U \in \mathbb{R}^{n \times k}$, $\Lambda \in \mathbb{R}^{k \times k}$ and link function $L(\cdot)$. Each node i thus has a corresponding *latent vector* $u_i \in \mathbb{R}^k$, where k is the number of *latent features*. In the setup of Equation 1, we have

$$\hat{G}_{ij}(U, \Lambda) = L(u_i^T \Lambda u_j).$$

The regularizer $\Omega(U, \Lambda) = \frac{\lambda_U}{2} ||U||_F^2 + \frac{\lambda_L}{2} ||\Lambda||_F^2$ usually. Such models have been successful in other dyadic prediction tasks such as collaborative filtering [20]. This approach has not been studied as extensively in the link prediction literature as it has in the collaborative filtering literature. Bayesian versions of these methods using a sigmoidal link function have been studied in statistics [16] and political science [34], where they are sometimes referred to as *bilinear random effects* models. These fields have focussed more on qualitative analysis of link behaviour than predictive performance and scalability. In the machine learning literature, computationally efficient frequentist versions of these latent feature models have been studied in [23,37], and Bayesian models have also been extended to allow for an infinite number of latent features [24,25].

2.1 Do Existing Methods Meet the Challenges in Link Prediction?

We recall the three challenges in link prediction from Section 1.1, and study how they are handled by current models.

– **Incorporating topological and side information.** Existing models typically use a nonlinear classifier such as a kernel SVM [14] or decision tree [21] to combine topological and explicit features. However, the topological structure is exploited by just learning weights on unsupervised scores. We will show that this is potentially limiting, since unsupervised scores have

limited expressivity. Latent feature methods like [16] do incorporate side information, but we will subsequently describe a limitation of this approach.
- **Overcoming imbalance.** Relatively little attention has been paid to modifying models so as to account for the class imbalance. One solution is undersampling the set of training dyads [14,21], but this has the disadvantage of necessarily throwing away information. [9] addresses imbalance by casting the problem as being cost-sensitive with unknown costs, and tunes the costs via cross-validation.
- **Scaling to large graphs.** Methods based on topological scores generally scale to large graphs, by virtue of mostly requiring only simple operations on the adjacency matrix. Some scores that look at long-range relationships between node pairs, such as the Katz measure, can be approximated in order to run on large graphs [10]. For methods based on explicit features, undersampling to overcome imbalance also reduces the number of training examples [21]. Latent class and latent feature methods based on Bayesian models do not scale to large graphs due to the cost of MCMC sampling.

We summarize representative link prediction methods in Table 1, and note whether they meet the three challenges. We would like a model that has the same expressiveness as existing methods, while directly addressing the above challenges. Our proposal is to extend matrix factorization to this end. The next section proposes the model, and argues why it is an suitable foundation for a link prediction model.

Table 1. A comparison of various link prediction methods. The *'s for "Large graphs?" indicate methods that perform some pre-processing on the data.

Class	Method	side information?	Imbalance?	Large graphs?
Unsupervised	Adamic-Adar [22]	No	No	Yes
	Katz [22]	No	No	Yes
Feature-based	Toplogical feats [14]	Optional	No	Yes
	CCP [9]	Optional	Yes	Yes*
	HPLP [21]	Optional	Yes	Yes*
Graph regularizer	LP [19]	Required	No	No
	ELP [26]	Required	No	Yes
Latent class	MMSB [3]	No	No	No
	IBP [24]	Optional	No	No
	IMRM [25]	Optional	No	No
Latent feature	Random effects [16]	Optional	No	No
	LFL [23]	Optional	No	Yes
	Our model	Optional	Yes	Yes

3 Extending Matrix Factorization for Link Prediction

A basic matrix factorization model for link prediction involves optimizing

$$\min_{U,\Lambda,b} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \ell(G_{ij}, L(u_i^T \Lambda u_j + b_i + b_j)) + \Omega(U,\Lambda) \tag{3}$$

for some appropriate link function $L(\cdot)$, loss function $\ell(\cdot)$ and regularizer $\Omega(\cdot, \cdot)$. As explained in the previous section, we interpret each $u_i \in \mathbb{R}^k$ as being a latent vector for each node, and so this is also called a *latent feature* approach. The b_i and b_j terms are node-specific *biases*, which are analogous to the intercept term in standard supervised learning. If the graph is undirected, then we can absorb Λ into the U matrix. For directed graphs, we can let Λ be an arbitrary asymmetric matrix, following [38,23].

3.1 Why is the Factorization Approach Appealing?

One nice property of the above model is that it can be trained using stochastic gradient descent, where we repeatedly draw a random $(i, j) \in \mathcal{O}$ and update u_i and u_j based on the corresponding gradients. A sweep over all observed (i, j) is known as an epoch, and often only a small constant number of epochs is needed for convergence. Training is thus linear in the number of observed dyads. As noted earlier, latent feature models for link prediction have been considered previously [16,34], but are typically trained with MCMC sampling, thus limiting analysis to small graphs. By contrast, with stochastic gradient descent we can handle graphs with several thousands of nodes and millions of edges.

Of course, scalability would be useless if the model were insufficiently rich. In fact, latent feature models can be seen as a generalization of latent class models, which may be thought of as learning a binary matrix $U \in \{0, 1\}^{n \times C}$, where C is the number of classes, and predicting UWU^T for a matrix W of inter-class link scores. Latent features can also be viewed as a much richer way of exploiting topological information than popular unsupervised measures described in the previous section. By construction, the latent feature approach exploits the graph topology so as to be maximally predictive of link behaviour. Thus in general, one would expect its scores to be more accurate than any single unsupervised method. A further conceptual advantage over unsupervised scores is that the learned u_i's let us make qualitative analyses of the nodes in the graph; for example, they may be used to visualize the structure of the graph.

Note that basic latent feature models are *not* the same as just computing the singular value decomposition (SVD) of the adjacency matrix with unknown status and known absent edges collapsed into a single class. Latent feature methods only attempt to model the known present and known absent edges in the graph, with regularization to prevent overfitting; no effort is spent in modelling the unknown status edges. The solutions of the two models are thus very different.

The other appealing property about matrix factorization is that there are intuitive ways to extend the model to incorporate side information and overcome the imbalance problem. We now describe these extensions in turn.

3.2 How Do We Combine Explicit and Latent Features?

Suppose we have explicit features $x_i \in \mathbb{R}^d$ for the ith row (or column) of the data, and features $z_{ij} \in \mathbb{R}^D$ for each dyad. A standard way to incorporate these features with the latent features is via a linear combination:

$$\min_{U,\Lambda,w,v,b} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \ell(G_{ij}, L(u_i^T \Lambda u_j + b_i + b_j + f_D(z_{ij}; w) + f_M(x_i, x_j; v))) + \Omega(U, \Lambda, w, v), \tag{4}$$

where $f_D(z_{ij}; w) = w^T z_{ij}$ and $f_M(x_i, x_j; v) = v^T x_i + v^T x_j$. This underpins the approaches of [24,16,15,23]. There is a subtle point regarding the choice of f_M. For the monadic features x_i, choosing $f_M(x_i, x_j; v) = v^T x_i + v^T x_j$ corresponds to forming a vector $x_{ij} = \begin{bmatrix} x_i & x_j \end{bmatrix}$ for the pair (i, j), and using a standard linear model. However, a drawback of this approach is that it only learns the *propensity* of i and j for the outcome G_{ij} [32]. Specifically, if we did not have the latent and dyadic features, for a fixed user i, the model would produce an identical ranking of link scores to every other user. This is obviously very restrictive.

An alternative is to use the prediction function $f_M(x_i, x_j; V) = x_i^T V x_j$, where $V \in \mathbb{R}^{d\times d}$. This does not suffer from the propensity issue, and is known as a *bilinear regression* model [11]. (To contrast, we will refer to the previous model as *unilinear* regression.) We let V be symmetric iff the graph G is. In a collaborative filtering context, such bilinear models have been used in [8,37]. Note that we need to learn d^2 parameters, which may be prohibitive. In such cases, we can either perform dimensionality reduction on the x_i's, or constrain V to be a diagonal plus low-rank matrix, $V = D + A^T B$, and learn the factors A, B, D.

The ability to augment latent with explicit features has a pleasant consequence, namely that we can combine latent features with the results of any other link prediction model. Suppose another model returns scores $\hat{G}_{ij}$ for the dyad (i, j). Then, we can treat this as being a dyadic feature z_{ij} in the above framework, and learn latent features that fit the residual of these scores. In general then, the latent feature approach has a natural mechanism by which any predictive signal can be incorporated, whether it is an explicit feature vector or model predictions. However, a caveat is in order: it is not necessary that combining latent features with another model will improve performance on test data. If the latent features learn similar structure to the other model, then combining the two cannot be expected to yield better results.

As a final remark, we note that the linear combination of latent and explicit features is not the only way to incorporate side information. This issue has been studied in the context of the *cold-start* problem [29] in collaborative filtering. Recent advances in this literature are based on inferring reasonable values of latent features by falling back to the side information as a prior [2,12]. However, unlike most collaborative filtering applications, in link prediction we are mostly interested in using side information to improve predictions, rather than dealing with cold-start nodes. Therefore, we expect it to be most useful to directly augment the latent feature prediction with one based on side information.

3.3 How Do We Overcome Imbalance?

Imbalanced classes pose a problem for at least two reasons: (i) with fewer examples of one class, it is more difficult to infer reliable patterns (ii) it is standard to train models to optimize for 0-1 accuracy, which can be made very high

by trivially predicting everything to belong to the dominant class; hence, most models are prone to yield biased results. Our matrix factorization model is not immune to these problems when trained with square- or log-loss, which may be seen as convex approximations to the 0-1 loss. Therefore, we must consider how to modify the model to account for imbalance. A standard strategy to overcome imbalance in supervised learning is undersampling [6], where we randomly throw out examples from the dominant class till the classes are more reasonably balanced. A disadvantage of undersampling is that it necessarily throws out information in the training set. Further, it is not clear to what ratio we can undersample without compromising the variance of our learned model.

An alternative is based on the following observation: in imbalanced classification problems, we often measure performance using AUC, which is agnostic to the distribution of classes. Intuitively, to get good test set AUC, it makes sense to *directly optimize* for AUC on the training set. This implicitly overcomes the imbalance problem, while simultaneously attempting to get the best AUC score on test data. This idea appears under-explored in the supervised learning literature, possibly due to the perceived complexity of optimizing for AUC. However, it is possible to optimize for AUC even on very large datasets [30]. To do this, we begin with the pairwise SVMRank framework [18]. Consider a binary classification scenario with training set $\{(x_i, y_i)\}$, and let $P = \{i : y_i = 1\}$ and $N = \{i : y_i = 0\}$. The empirical AUC of a linear classifier with weight w is

$$\hat{\mathcal{A}} = \frac{1}{|N||P|} \sum_{i \in P} \sum_{j \in N} \mathbf{1}[w^T x_i > w^T x_j].$$

The problem of maximizing AUC can be cast as

$$\min_w \frac{1}{|N||P|} \sum_{i \in P} \sum_{j \in N} \mathbf{1}[w^T (x_i - x_j) < 0].$$

The above is a binary classification task on $\{(x_i - x_j, 1)\}_{(i,j) \in \mathcal{Q}}$, where $\mathcal{Q} = \{(i, j) : y_i = 1, y_j = 0\}$. Thus, to maximize the AUC, we can replace the indicator function above with a regularized loss function:

$$\min_w \frac{1}{|N||P|} \sum_{(i,j) \in \mathcal{Q}} \ell(w^T (x_i - x_j), 1) + \Omega(w).$$

The above can be optimized efficiently using stochastic gradient descent, where at each iteration we randomly pick a *pair* of examples and compute the gradient with respect to them [30]. We can directly translate this idea to matrix factorization for link prediction. However, there is a subtlety in how we decide what pairs of examples to consider. Suppose $\mathcal{O}^+, \mathcal{O}^-$ are the sets of known present and known absent dyads respectively. Then, there are two ways in which we can construct pairs of nodes.

Per-node pairs. Here, we consider (known present, known absent) pairs that share one node in common. This can be thought as applying the AUC loss for every row of the G matrix. The optimization is

$$\min_{U,\Lambda} \frac{1}{|\mathcal{D}|} \sum_{i=1}^{n} \frac{1}{|\mathcal{O}_i^+||\mathcal{O}_i^-|} \sum_{j \in \mathcal{O}_i^+, k \in \mathcal{O}_i^-} \ell(u_i^T \Lambda (u_j - u_k), 1) + \Omega(U),$$

where $\mathcal{D}$ is set of all (i, j, k) triplets where $j \in \mathcal{O}_i^+, k \in \mathcal{O}_j^-$. In the case of logistic loss, the above model is similar to BPR [27], although the motivations of the two models are different: BPR was proposed to deal with implicit feedback (or positive only) collaborative filtering datasets. In the above, we are treating the known present links as the "positive feedback", and only links in $\mathcal{O}_i^-$ as being "unspecified feedback". Further, we combine the learned latent features with side information via bilinear regression.

All pairs. Here, we consider (known present, known absent) pairs that *need not* share a node in common. This can be thought as applying the AUC loss globally on every dyad in G. The optimization is

$$\min_{U,\Lambda} \frac{1}{|\mathcal{O}^+||\mathcal{O}^-|} \sum_{(i,j) \in \mathcal{O}^+, (i',j') \in \mathcal{O}^-} \ell(u_i^T \Lambda u_j - u_{i'}^T \Lambda u_{j'}, 1) + \Omega(U).$$

The choice between the two schemes depends on whether we ultimately evaluate AUC on a per-node or global basis. This choice in turn is largely problem specific; for example, in a social network we would like each individual user to have a good ranking, whereas in a protein-protein interaction network, our interest is in which unobserved dyads are worth performing further analysis on. Regardless of the choice of scheme, we are faced with the problem of having to learn from a large number of examples, one that is superlinear in the number of observed dyads. However, in practice, only a fraction of a single epoch of stochastic gradient descent is needed to achieve good results; we will discuss this more in our experiments.

3.4 The Final Model

Our final model optimizes for AUC directly, with regularization to prevent overfitting. We linearly combine side information via a bilinear regression model. Assuming we follow the per-node ranking scheme, and assuming per-node and per-dyad side information x_i and z_{ij} respectively, the objective is:

$$\min_{U,\Lambda,V,w,b} \frac{1}{|\mathcal{O}|} \sum_{i=1}^{n} \sum_{j \in \mathcal{O}_i^+, k \in \mathcal{O}_i^-} \ell(L(u_i^T \Lambda (u_j - u_k) + b_i + b_j + x_i^T V x_j + w^T z_{ij}), 1) +$$
$$\frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_\Lambda}{2}||\Lambda||_F^2 + \frac{\lambda_V}{2}||V||_F^2 + \frac{\lambda_w}{2}||w||_2^2, \quad (5)$$

where the link function $L(\cdot)$ and loss $\ell(\cdot, \cdot)$ are user-specified, and $\Lambda = I$ if the graph is symmetric. Further, if required, we factorize the bilinear weights $V = D + A^T B$ for diagonal D and arbitrary A, B, and learn A, B, D.

4 Experimental Design

We present an experimental comparison of our model to other link prediction methods. We used datasets from a range of applications of link prediction:

- **Computational biology**
  - Protein-protein interaction data from [31], denoted "Prot-Prot". Each protein has a 76 dimensional feature vector.
  - Metabolic pathway interaction data for *S. cerevisiae* provided in the KEGG/PATHWAY database [36], denoted "Metabolic". Each node has three sets of features: a 157 dimensional vector of phylogenetic information, a 145 dimensional vector of gene expression information, and a 23 dimensional vector of gene location information.
- **Bibliographic networks**
  - The co-author network of authors at the NIPS conference [28], denoted "NIPS". Each node has a 14035 dimensional bag-of-words feature vector, being the words used by the author in her publications. We performed latent semantic indexing (LSI) to reduce the number of features to 100.
  - The co-author network of condensed-matter physicists [21], denoted "Condmat". Following [21], we consider node-pairs that are 2 hops away in the first five years of the data. There is no side information in this problem.
- **Other networks**
  - A network of military disputes between countries [13], denoted "Conflict". Following [34], we considered all disputes in the period 1990–2000. The graph is directed, with an edge originating from the conflict initiator. Due to time constraints, we only report results on the symmetrized version of the data, whether two countries have a link if either initiated conflict with the other. Each node has 3 features, comprising the country's population, GDP and polity, and additionally each dyad has 6 features, including e.g. the countries' geographic distance.
  - The US electric powergrid network [35], denoted "PowerGrid". There is no side information in this problem. This dataset is challenging because of the extreme imbalance (1 link for every 1850 non-links), and the nature of its linking behaviour: we expect two nodes to be linked if they are nearby geographically, which is a latent feature that may be difficult to infer.

To evaluate the models, we kept aside a fixed fraction of the observed dyads $\mathcal{O}$ for training various models, and evaluated AUC on a test set comprising the remaining dyads. This process was repeated 10 times, and we report the average test set AUC. We used two training split ratios: for the datasets with side information (Prot-Prot, Metabolic, NIPS and Conflict), we used 10% of dyads for training, and for the others (Condmat, PowerGrid), we used 90% of dyads for training. On the latter two datasets a 10% split would cause most nodes to have no known present links in the training set, making it difficult to make predictions based solely on the topological structure.

Table 2. Properties of datasets used in experimental comparison

Dataset	Nodes	$\|\mathcal{O}^+\|$	$\|\mathcal{O}^-\|$	+ve:−ve ratio	Average degree
Prot-Prot	2617	23710	6,824,979	1 : 300	9.1
Metabolic	668	5564	440,660	1 : 80	8.3
NIPS	2865	9466	8,198,759	1 : 866	3.3
Condmat	14230	2392	429,232	1 : 179	0.17
Conflict	130	320	16580	1 : 52	2.5
PowerGrid	4941	13188	24,400,293	1 : 1850	2.7

We swept over a grid of regularization parameters and learning rates for stochastic gradient descent, and evaluated the average AUC across the 10 splits. We report results for the parameters selected by the grid search. The number of latent features, k, was informally picked to be 30 for all datasets except Conflict, where only $k = 5$ were needed to achieve good performance. Our MATLAB scripts are available for download at `http://cseweb.ucsd.edu/~akmenon/code`.

Table 2 summarizes the dataset sizes in terms of number of nodes and known present/known absent dyads. Condmat is the only dataset where some edges have genuinely missing status even at test time (corresponding to node pairs more than 2 hops away). Note that we do not undersample any of the datasets. We see that PowerGrid is the most imbalanced dataset, while also having a small average degree.

5 Experimental Results

We divide the experimental results into three parts, each considering a different type of method. Methods with scores in **bold** have the highest score amongst methods being compared in that table. Methods that additionally have a **star** * are the best performing across all tables. In both cases, we only consider differences in AUC that are greater than the standard deviation across the splits.

Do latent features improve on unsupervised scores? We first report results on the following methods, which only exploit topological information:

- *Unsupervised scores.* We used Adamic-Adar (AA), Preferential Attachment (PA), Shortest-Path (SHP) and Katz, which are popular scores that perform well on a range of graphs [22]. We also ran linear regression on all unsupervised scores (Sup-Top), which attempts to find a weighted combination of the scores with better performance.
- *Raw SVD.* We computed the raw SVD on the adjacency matrix, treating known absent and unknown status edges as being one and the same.
- *Factorization.* We used the factorization model of Equation 3 using square-loss and identity link (Fact-Sq), and log-loss with sigmoid link (Fact-Log).
- *Unsupervised scores as input to factorization.* Here, we used all the unsupervised scores as features for each dyad, and fed them into a factorization model trained with square-loss (Fact+Scores).

Table 3. Test AUC scores for methods based on topological features alone

Dataset	AA	PA	SHP	Katz	Sup-Top
Prot-Prot	0.564 ± 0.005	0.750 ± 0.003	0.726 ± 0.005	0.727 ± 0.005	0.754 ± 0.003
Metabolic	0.524 ± 0.005	0.524 ± 0.005	0.626 ± 0.004	0.608 ± 0.007	0.628 ± 0.001
NIPS	0.512 ± 0.002	0.543 ± 0.005	0.517 ± 0.003	0.517 ± 0.003	0.542 ± 0.007
Condmat	0.567 ± 0.014	0.716 ± 0.026	0.673 ± 0.018	0.673 ± 0.017	0.720 ± 0.020
Conflict	0.507 ± 0.008	0.546 ± 0.024	0.512 ± 0.014	0.512 ± 0.014	$\mathbf{0.695 \pm 0.076}$
PowerGrid	0.589 ± 0.003	0.442 ± 0.010	0.659 ± 0.015	0.655 ± 0.016	$\mathbf{0.708 \pm 0.062}^*$

Dataset	SVD	Fact-Sq	Fact-Log	Fact+Scores
Prot-Prot	0.635 ± 0.003	$\mathbf{0.795 \pm 0.005}$	$\mathbf{0.793 \pm 0.002}$	$\mathbf{0.793 \pm 0.005}$
Metabolic	0.538 ± 0.017	$\mathbf{0.696 \pm 0.001}$	$\mathbf{0.695 \pm 0.001}$	$\mathbf{0.696 \pm 0.002}$
NIPS	0.512 ± 0.031	$\mathbf{0.612 \pm 0.007}$	$\mathbf{0.610 \pm 0.008}$	$\mathbf{0.613 \pm 0.019}$
Condmat	0.629 ± 0.051	$\mathbf{0.810 \pm 0.020}^*$	$\mathbf{0.822 \pm 0.025}^*$	$\mathbf{0.812 \pm 0.020}^*$
Conflict	0.541 ± 0.094	$\mathbf{0.692 \pm 0.040}$	$\mathbf{0.692 \pm 0.039}$	$\mathbf{0.689 \pm 0.042}$
PowerGrid	0.691 ± 0.026	0.637 ± 0.012	0.675 ± 0.017	$\mathbf{0.751 \pm 0.020}^*$

Our results are given in Table 3. (The table is split into two halves for visual clarity.) We make the following observations:

- Individual unsupervised scores are always outperformed by factorization methods by around 5%–25%. In most cases, factorization also outperforms a supervised combination of such scores. This suggests that in general, latent features better exploit topological information by virtue of directly optimizing to be predictive of link behaviour.
- In some cases, combining multiple topological features worsens test set AUC. The reason is likely twofold: first, a linear combination of weights may be too simplistic to leverage their combined power. Second, linear regression may underperform due to the imbalance of the data, as noted in Section 3.3.
- In most cases, combining latent features and unsupervised scores does not improve performance. This indicates that the unsupervised scores do not capture sufficiently complementary information to the latent features.
- As conjectured in Section 3.1, raw SVD performs much worse than the factorization methods on the datasets using 10% of dyads for training, as it treats all missing edges as being known absent.
- Amongst the two factorization approaches, the choice of loss function generally does not influence results significantly. For both losses, we required no more than 10 epochs to converge on any dataset.

How predictive is side information? Next, we tried several methods that use side information $x_i \in \mathbb{R}^d$ for each node i:

- *Raw similarity.* As a baseline, we use the cosine similarity $\frac{x_i^T x_j}{||x_i|| ||x_j||}$ as the predicted score for the node-pair (i, j).
- *Link propagation.* We use Link Propagation (LP) with the "sum kernel" as defined in [19], using cosine similarity as our base measure. We specifically use a special case of LP described in [19] that can be efficiently implemented in MATLAB using Lyapunov functions. We also tried an approximation to this method, Exact Link Propagation (ELP) [26].

Table 4. Test AUC scores for methods based on explicit features. Condmat and PowerGrid are not included because they do not have side information.

Dataset	Similarity	LP	ELP	ULR	BLR	Fact+LP	Fact+BLR
Prot-Prot	0.680 ± 0.002	0.771 ± 0.002	0.740 ± 0.003	0.670 ± 0.002	0.776 ± 0.006	0.789 ± 0.003	$\mathbf{0.813 \pm 0.002}^*$
Metabolic	0.605 ± 0.002	0.719 ± 0.001	0.659 ± 0.010	0.694 ± 0.007	0.725 ± 0.012	0.701 ± 0.002	$\mathbf{0.763 \pm 0.006}^*$
NIPS	$\mathbf{0.953 \pm 0.000}$	0.767 ± 0.004	0.929 ± 0.010	0.611 ± 0.007	$\mathbf{0.951 \pm 0.002}$	0.885 ± 0.032	0.945 ± 0.003
Conflict	0.577 ± 0.008	0.614 ± 0.016	0.648 ± 0.029	$\mathbf{0.869 \pm 0.029}^*$	$\mathbf{0.891 \pm 0.017}^*$	0.693 ± 0.046	$\mathbf{0.890 \pm 0.017}^*$

- *Regression.* We apply unilinear (ULR) and bilinear (BLR) regression on the feature vectors, corresponding to Equation 4 with the two choices of prediction function $f_M(\cdot, \cdot)$ discussed in Section 3.2, and where the latent features are omitted. Both methods did *not* use unsupervised scores as input, and were trained with square-loss.
- *Combinations.* We combined the factorization model with Link Propagation (Fact+LP) and bilinear regression (Fact+BLR), the latter being the model given in Equation 4 for bilinear $f_M(\cdot, \cdot)$.

We present the results in Table 4. (Condmat and PowerGrid are not included because they do not possess side information.) We observe the following:

- Bilinear regression is better than plain factorization on all datasets but Prot-Prot, which indicates that it is difficult to infer latent structure from the observed data that is more predictive than the given side information. This is not surprising given the sparsity of known present edges in the datasets.
- On Prot-Prot and Metabolic, jointly learning latent features and a bilinear regression model gives better performance than doing either individually. This suggests that despite the general superiority of explicit over latent features, the two can have complementary characteristics.
- The factorization model does not benefit from incorporating the output of LP. In fact, we find the test set AUC decreases on the NIPS dataset. On most datasets, LP had training AUC close to 1, suggesting that it is difficult to learn latent features on top of these scores without overfitting.
- Unilinear regression is always outperformed by bilinear regression, usually by a significant margin. This shows that the propensity problem with unilinear regression, discussed in §3.2, has important practical implications.
- Bilinear regression always outperforms both variants of Link Propagation.

Does optimizing for a ranking loss overcome imbalance? Finally, we check whether directly optimizing for AUC helps overcome imbalance. We apply the model of Equation 5 using square-loss (Fact-Rank), and consider an alternative where the ranking is defined over all dyads (Fact-Rank-Global) (see Section 3.3 regarding the per-node versus global ranking). We also optimize the BLR and Fact+BLR models of the previous section with the per-node form of ranking loss (BLR-Rank and Fact+BLR-Rank). On datasets with many dyads, the ranking losses only required a small fraction of a single epoch to converge (e.g.

Table 5. Test AUC scores for methods optimized with ranking loss

Dataset	Fact-Rank	Fact-Rank-Global	BLR-Rank	Fact+BLR-Rank
Prot-Prot	0.798 ± 0.001	0.794 ± 0.001	0.785 ± 0.003	$\mathbf{0.806 \pm 0.003}$
Metabolic	0.705 ± 0.007	0.706 ± 0.006	$\mathbf{0.764 \pm 0.007}^*$	$\mathbf{0.765 \pm 0.007}^*$
NIPS	0.609 ± 0.008	0.605 ± 0.007	0.949 ± 0.002	$\mathbf{0.956 \pm 0.002}^*$
Condmat	$\mathbf{0.814 \pm 0.019}^*$	$\mathbf{0.826 \pm 0.019}^*$	N/A	N/A
Conflict	0.690 ± 0.042	0.686 ± 0.042	$\mathbf{0.885 \pm 0.018}^*$	$\mathbf{0.886 \pm 0.021}^*$
PowerGrid	0.723 ± 0.015	$\mathbf{0.754 \pm 0.014}^*$	N/A	N/A

on PowerGrid, 1%–5% of an epoch for per-node ranking, and 0.01%–0.05% for global ranking). From the results in Table 5, we note that:

- For factorization methods, the ranking loss is dramatically superior to the regression losses on the PowerGrid dataset, which is the most imbalanced and has a small average degree. On other datasets, the differences are more modest, but the ranking loss is always competitive with the regression losses, while requiring significantly fewer dyads for convergence.
- For bilinear regression, optimizing with a ranking loss gives better performance than square loss on Prot-Prot and Metabolic.
- Jointly learning latent features and a bilinear regression model with a ranking loss performs at least as well as optimizing with square loss.

6 Conclusion

In the paper, we proposed a model that extends matrix factorization to solve structural link prediction problems in (possibly directed) graphs. Our model combines *latent features* with optional explicit features for nodes and edges in the graph. The model is trained with a *ranking loss* to overcome the imbalance problem that is common in link prediction datasets. Training is performed using stochastic gradient descent, and so the model scales to large graphs. Empirically, we find that the latent feature approach significantly outperforms popular unsupervised scores, such as Adamic-Adar and Katz. We find that it is possible to learn useful latent features on top of explicit features, which can give better performance than either model individually. Finally, we observe that optimizing with a ranking loss can improve AUC performance by around 10% over a standard regression loss. Overall, on six datasets from widely different domains, some possessing side information and others not, our proposed method (Fact-BLR-Rank from Table 5 on datasets with side information, Fact-Rank on the others) has equal or better AUC performance (within statistical error) than previously proposed methods.

References

1. Adamic, L.A., Adar, E.: Friends and neighbors on the web. Social Networks 25(3), 211–230 (2003)
2. Agarwal, D., Chen, B.-C.: Regression-based latent factor models. In: KDD 2009, pp. 19–28. ACM, New York (2009)
3. Airoldi, E., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. In: NIPS, pp. 33–40 (2008)
4. Batagelj, V., Ferligoj, A., Doreian, P.: Generalized blockmodeling. Informatica (Slovenia) 23(4) (1999)
5. Beck, N., King, G., Zeng, L.: Improving quantitative studies of international conflict: A conjecture. American Political Science Review 94(1), 21–36 (2000)
6. Chawla, N.V., Japkowicz, N., Kotcz, A.: Editorial: special issue on learning from imbalanced data sets. SIGKDD Explor. Newsl. 6, 1–6 (2004)
7. Chen, H., Li, X., Huang, Z.: Link prediction approach to collaborative filtering. In: Joint Conference on Digital Libraries, vol. 7, pp. 141–142 (2005)
8. Chu, W., Park, S.-T.: Personalized recommendation on dynamic content using predictive bilinear models. In: WWW 2009, pp. 691–700. ACM, New York (2009)
9. Doppa, J.R., Yu, J., Tadepalli, P., Getoor, L.: Learning algorithms for link prediction based on chance constraints. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6321, pp. 344–360. Springer, Heidelberg (2010)
10. Dunlavy, D.M., Kolda, T.G., Acar, E.: Temporal link prediction using matrix and tensor factorizations. ACM Transactions on Knowledge Discovery from Data (in Press, 2011)
11. Ruben Gabriel, K.: Generalized bilinear regression. Biometrika 85, 689–700 (1998)
12. Gantner, Z., Drumond, L., Freudenthaler, C., Rendle, S., Schmidt-Thieme, L.: Learning attribute-to-feature mappings for cold-start recommendations. In: ICDM, pp. 176–185 (2010)
13. Ghosn, F., Palmer, G., Bremer, S.: The MID3 data set, 1993-2001: Procedures, coding rules, and description. Conflict Management and Peace Science 21, 133–154 (2004)
14. Hasan, M.A., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: SDM Workshop on Link Analysis, Counterterrorism and Security (2006)
15. Hoff, P.: Modeling homophily and stochastic equivalence in symmetric relational data. In: NIPS (2007)
16. Hoff, P.D.: Bilinear mixed effects models for dyadic data. Journal of the American Statistical Association 32, 100–286 (2003)
17. Hofmann, T., Puzicha, J., Jordan, M.I.: Learning from dyadic data. In: NIPS II, pp. 466–472. MIT Press, Cambridge (1999)
18. Joachims, T.: Optimizing search engines using clickthrough data. In: KDD 2002, pp. 133–142. ACM, New York (2002)
19. Kashima, H., Kato, T., Yamanishi, Y., Sugiyama, M., Tsuda, K.: Link propagation: A fast semi-supervised learning algorithm for link prediction. In: SDM, pp. 1099–1110 (2009)
20. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42, 30–37 (2009)
21. Lichtenwalter, R., Lussier, J.T., Chawla, N.V.: New perspectives and methods in link prediction. In: KDD, pp. 243–252 (2010)

22. Lu, L., Zhou, T.: Link prediction in complex networks: A survey (2010), http://arxiv.org/abs/1010.0725
23. Menon, A.K., Elkan, C.: A log-linear model with latent features for dyadic prediction. In: ICDM (2010)
24. Miller, K., Griffiths, T., Jordan, M.: Nonparametric latent feature models for link prediction. In: NIPS, vol. 22, pp. 1276–1284 (2009)
25. Mørup, M., Schmidt, M.N., Hansen, L.K.: Infinite multiple membership relational modeling for complex networks. In: NIPS Workshop on Networks Across Disciplines in Theory and Application (2010)
26. Raymond, R., Kashima, H.: Fast and scalable algorithms for semi-supervised link prediction on static and dynamic graphs. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6323, pp. 131–147. Springer, Heidelberg (2010)
27. Rendle, S., Freudenthaler, C., Gantner, Z., Lars, S.-T.: BPR: Bayesian personalized ranking from implicit feedback. In: UAI 2009, pp. 452–461. AUAI Press, Arlington (2009)
28. Roweis, S.: NIPS dataset (2002), http://www.cs.nyu.edu/~roweis/data.html
29. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: SIGIR 2002, pp. 253–260. ACM, New York (2002)
30. Sculley, D.: Large scale learning to rank. In: NIPS Workshop on Advances in Ranking (2009)
31. Tsuda, K., Noble, W.S.: Learning kernels from biological networks by maximizing entropy. Bioinformatics 20, 326–333 (2004)
32. Vert, J.-P., Jacob, L.: Machine learning for in silico virtual screening and chemical genomics: New strategies. Combinatorial Chemistry & High Throughput Screening 11(8), 677–685 (2008)
33. Wang, C., Satuluri, V., Parthasarathy, S.: Local probabilistic models for link prediction. In: ICDM, pp. 322–331 (2007)
34. Ward, M.D., Siverson, R.M., Cao, X.: Disputes, democracies, and dependencies: A reexamination of the Kantian peace. American Journal of Political Science 51(3), 583–601 (2007)
35. Watts, D.J., Strogatz, S.H.: Collective dynamics of "small-world" networks. Nature 393(6684), 440–442 (1998)
36. Yamanishi, Y., Vert, J.-P., Kanehisa, M.: Supervised enzyme network inference from the integration of genomic data and chemical information. In: ISMB (Supplement of Bioinformatics), pp. 468–477 (2005)
37. Yang, S.H., Long, B., Smola, A., Sadagopan, N., Zheng, Z., Zha, H.: Like like alike – joint friendship and interest propagation in social networks. In: WWW (2011)
38. Zhu, S., Yu, K., Chi, Y., Gong, Y.: Combining content and link for classification using matrix factorization. In: SIGIR 2007, pp. 487–494. ACM, New York (2007)
39. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Technical report, CMU (2002)

On Oblique Random Forests

Bjoern H. Menze[1,2], B. Michael Kelm[1], Daniel N. Splitthoff[1], Ullrich Koethe[1], and Fred A. Hamprecht[1]

[1] Interdisciplinary Center for Scientific Computing, University of Heidelberg, Germany
[2] Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, USA

Abstract. In his original paper on random forests, Breiman proposed two different decision tree ensembles: one generated from "orthogonal" trees with thresholds on individual features in every split, and one from "oblique" trees separating the feature space by randomly oriented hyperplanes. In spite of a rising interest in the random forest framework, however, ensembles built from orthogonal trees (RF) have gained most, if not all, attention so far.

In the present work we propose to employ "oblique" random forests (oRF) built from multivariate trees which explicitly learn optimal split directions at internal nodes using linear discriminative models, rather than using random coefficients as the original oRF. This oRF outperforms RF, as well as other classifiers, on nearly all data sets but those with discrete factorial features. Learned node models perform distinctively better than random splits. An oRF feature importance score shows to be preferable over standard RF feature importance scores such as Gini or permutation importance. The topology of the oRF decision space appears to be smoother and better adapted to the data, resulting in improved generalization performance. Overall, the oRF propose here may be preferred over standard RF on most learning tasks involving numerical and spectral data.

1 Introduction

Random forests have gained popularity in high-dimensional and ill-posed classification and regression tasks, for example on micro-arrays [19], time series [37], or spectral data [39,29], but also for inference in application such as image segmentation or object recognition in computer vision [9,45]. Random forests are comparable in performance to many other non-linear learning algorithms. They often do well with little parameter tuning [16], and are able to identify relevant feature subsets even in the presence of a large amount of irrelevant predictors [6,21,25,2]. More recently, additional properties of the random forest have gained interest, for example in feature selection [25,34,23,44] or the explorative analysis of sample proximities [38].

The main idea of the random forest framework, proposed by Breiman in [5], is to learn many variable but unbiased base learners, and to reduce variance by pooling over a whole committee of predictors. This concept is familiar from

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 453–469, 2011.

bagging [3], when a large number of decision trees is learned from random subsets of the training samples and their decisions are averaged in prediction. In random forests the correlation between individual base learners is further reduced by seeking at every node for the best prediction in a random subspace of the training data, similar to ideas from "random subspaces" [17] and "random splits" [10].

A random forest can be generated from two different kinds of trees. *Univariate decision trees* – such as CART or C4.5 – serve as base learners of the most popular random forest implementations. They separate feature space by hyperplanes that are *orthogonal* to single feature axes, resulting in the typical stair- or box-like decision surfaces of these classifiers. While this characteristic shape of the decision boundary might be advantageous for some data, one may argue that it is suboptimal for other – potentially leading to a substantial bias of the base learner. Collinear data with correlated features [23], for example, arising from spectra, time series, but also micro-arrays or image patches, may reside in subspaces that lie between the coordinate axes. In that case, class distributions may appear inseparable when marginal distributions are evaluated in the search for the best univariate split, and separating classes may require complex and deeply nested trees (Fig. 1). *Multivariate decision trees* – trees using decision surfaces at arbitrary, *oblique* orientations to the axes – may be better adapted to decisions in such subspaces [27], leading to decision boundaries that are less biased by geometrical constraints of the base learner. In his original paper [5], Breiman proposed the use of decision trees with oblique splits at random orientations, observing that this approach yielded "results never reached so far" on the data sets tested. Unlike their counterparts with univariate node models, however, these random forests have not met a wider interest, yet.

In the present work we propose to use random forests with regularized *oblique* model trees as base learners. Recent results on ensemble pruning showed an advantage of choosing base learners which are optimally adapted to the data [22]. So, rather than choosing *random* recursive splits, as Breiman suggests in [5], we focus on trees with *task optimal* recursive partitioning. In this we follow the idea also used, for example, in probabilistic boosting trees [43], which use "strong" discriminative model at each node to obtain overall better decision rules. Our approach is also related to "rotation forests" [33] where oblique split directions are sought from the principal components of feature subsets of the training data, reportedly improving results significantly in selected classification tasks. We differ from this approach by using supervised approaches to define optimal split directions. While we follow some of the ideas of [40,41], we refrain from using global optimization techniques to train the oblique tree. Furthermore, we also perform experiments to understand the benefit of learned oblique node models. For splitting feature space at a node, a wide range of linear discriminative node models can be used, all employing somewhat different optimization objectives. We emphasize on regularized node models which may complement the good properties of the original random forest by Breiman [5] for classifying high-dimensional data with few samples – a domain where the random forest with orthogonal trees performs exceptionally well. Finally, we propose feature

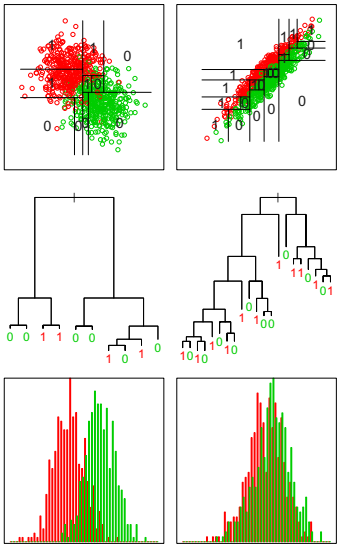


Fig. 1. Separation of correlated data using univariate decision trees. The left column shows a synthetic binary data set, the right column shows the same data after the samples have been artificially correlated. The top row shows the bivariate samples, and a segmentation using an orthogonal decision tree (black lines, and segments indicated by 0 and 1). The classification tree is visualized in the middle row. Marginal distributions of the horizontal axis (top row) are shown as histograms in the bottom row – these are the distributions evaluated at every node in a univariate decision tree. While the initial segmentation problem (left column) is simple, the strong overlap in the marginal distributions (right column) leads to highly nested decision rules and complex decision boundaries if features are correlated.

importance and visualization measures for the oblique random forest, similar to the ones proposed [5].

In the following we will shortly outline model-based *oblique random forests* (oRF) (Section 2). In subsequent experiments we will compare the oRF quantitatively against standard random forests with univariate trees (RF) and other related non-linear classifiers (Section 3), and try to understand properties of oblique random forests with learned node model (Section 4). We will finally propose feature importance and sample proximity measures derived from the oblique random forest (Section 5).

2 Oblique Random Forests

The oRF shares the ensemble generation processes with the "standard" RF [5]: For each tree a new set of samples is drawn randomly from the training data with replacement. At every recursive binary split, a new set of m_{try} features is sampled without replacement, and the optimal split is sought in the subspace spanned by these features. Differences to the standard procedure apply in the way optimal splits direction are sought at each node.

Oblique Model Trees. For oRF, we rely on multivariate models for binary splits in each node. For a sample $\mathbf{x} = [x_1, \ldots, x_{m_{try}}]^T$ in a m_{try}-dimensional space, the decision f at the node m can be formulated as:

$$f_m(\mathbf{x}) : \beta_m^T \mathbf{x} > c_m \tag{1}$$

with coefficients β_m defining the projection for the split and threshold c_m. Inferring the optimal β_m is more difficult than the identification of a single optimal

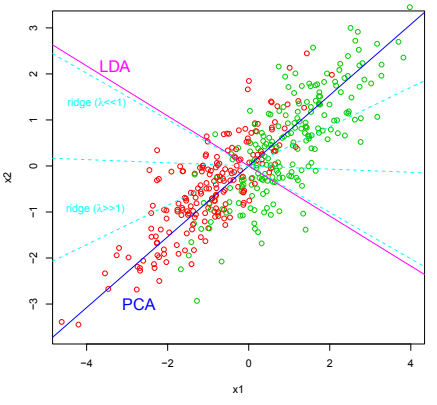


Fig. 2. Effect of regularization on split directions of the oblique node models. Lines represent normals of the discriminating hyperplanes. PCA seeks for oblique splits with maximal data support (first component shown here) and LDA for correlation with the class label. By using regularized regression a multitude of alternative subspaces with intermediate properties can be defined and evaluated. Regularization biases the optimal LDA-like projection towards one which has higher data-support, more directing towards the PCA-like solution.

feature and an appropriate threshold for a split in the univariate case. Different criteria can be used to find the linear subspace (Fig. 2). Projections for linear splits may consider class label information only (as in logistic regression or linear discriminant analysis), they may align with the data variation (as with principal component analysis), or they may seek for an optimum in between, trading class label correlation and data support (as in ridge or partial least squares regression, or linear support vector machines). Constrained regression methods also enjoying popularity in the classification of high dimensional data. They perform well in tasks with less observations than predictors, and they may help to find splits when less than m_{try} samples reside in a node which often occurs in deep split nodes far from the root. We therefore choose ridge regression for our split model:

$$\beta_{ridge}(\lambda) \sim \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{2} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{P} |\beta_j|^2. \tag{2}$$

using regularization parameter λ. With this choice the node model is optimizing for [11]

$$\beta_{ridge}(\lambda') \sim \underset{||\beta=1||}{\operatorname{argmax}} corr^2(\beta X, Y) * \frac{var(\beta X)}{var(\beta X) + \lambda'}. \tag{3}$$

The regularization parameter λ allows the classifier to adapt to an optimal split direction β_{ridge} in between two extremes (Fig. 2): With $\lambda = 0$, β_{ridge} may point towards maximal correlation with class labels, similar to linear discriminant analysis (LDA). With $\lambda \gg 1$, it may point towards highest data variation and data support, similar to principal component analysis (PCA).

At each node m, the model parameter λ can be adapted to the out-of-bag samples available at that node. Then all samples X_m are projected into β_m and the optimal split-point c_m on the scores $s_m = X_m\beta_m$ is identified using the Gini impurity $I(s_m) = \frac{1}{2} * (1 - \sum_{k=0,1} P_k^2(s_m))$. The threshold c_m maximizing the decrease in Gini impurity (i.e., the minimum in $I(s_m < c_m) + I(s_m > c_m)$) is

chosen and samples are separated accordingly. For both subsets the process of finding the best split is recursively iterated until both classes are separated.

Implementation of the Oblique Random Forest. In the random forest framework a large number of trees are combined. Two hyper-parameters control the generation of the decision trees in this ensemble: subspace dimensionality m_{try} and ensemble size n_{tree}. Parameter m_{try} determines the number of features sampled randomly at each individual node and the degree of randomness in the model generation. Parameter m_{try} has to be chosen to obtain a "sufficient" variability of the trees in the forest, ensuring that correlation between individual trees is as low as possible. In prediction new samples are pushed down each of the n_{tree} trees and are assigned the label in the terminal note and decisions can be pooled according to different schemes [32,30]. Here we use the normalized number of votes, or probability $p \in [0, 1]$, which is relatively robust against over-fitting [5,35,32].

We implement two versions of the oRF in our experiment: a) *oRF-ridge* optimizing regularization parameter λ at every split, b) *oRF-lda* performing an unregularized LDA-like split at every node, and c) *oRF-rnd* with random oblique splits as proposed in [5]. Trees are generated as follows[1]: 1) For each tree a new set of samples is drawn randomly from the training data with replacement. 2) Random subspaces of dimension m_{try} are sampled without replacement for every node. 3) At every split, we scale variables to zero mean and unit variance to enhance the stability of the linear model. 4a) For *oRF-ridge*, ridge regression is tested using $\lambda = 10^i$ with $i = \{-5, -4, \ldots, 5\}$, and λ is optimized using the out-of-bag samples residing at the same node. 4b) For *oRF-lda* we set $\lambda = 0$ and use the resulting node model. 4c) For *oRF-rnd* we draw random values from a normal distribution (with zero mean and standard deviation of one) to obtain coefficients β of the node model beta under a similar prior distribution as in a L_2 constrained ridge regression. For all three oRF, samples are projected into subspace determined by the node model. 5) Optimal thresholds for splitting the data are sought using the Gini impurity on the fitted scores of the training data, and the samples are split accordingly. For each of the n_{tree} trees, steps 2)-5) are repeated until all classes are separated (oRF-lda, oRF-rnd), or no out-of-bag test samples are available for adapting λ any more (oRF-ridge).

3 Comparison of Classification Performances

In a first experiment we compare the performance of oRF, RF and other learning methods on different types of data to identify and compare specific properties of these classifiers.

Experimental data. A number of binary benchmark problems are used for the evaluation: binary classification problems of the UCI data repository (data sets 1-15, Table 1), synthetic data sets (16-20) [4], binary groupings of handwritten

[1] Classifier publically available at `cran.r-project.org` in package *obliqueRF*.

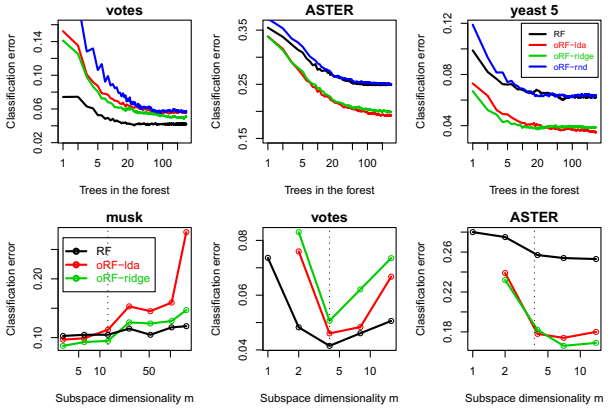


Fig. 3. Parametrization of the random forest. On most data sets default ensemble parameters for ensemble size n_{tree} (top) and subspace dimensionality m_{try} (indicated by the dotted vertical line, bottom) perform reasonably well.

digits (21-23) from the MNIST data base, binary data from detection problems in archaeological remote sensing (24-25) [26], from the analysis of magnetic resonance spectra for the detection of brain tumors or yeast infections (26-28, 32-36) [24,23], from the analysis of infrared spectra for BSE detection from blood serum and food analysis (29-31, 37-40) [25,23].

Alternative Classifiers. We choose seven alternative nonlinear classifiers for our comparison – most of them with some conceptual relation to oblique random forests. We test two other decision tree ensembles (Table 1): *adaboost* [12], the standard random forest (*RF*) [20] and a version of the extremely random forests [13] with $m_{try} = 1$ (*RF-rnd*), all relying on univariate decision trees. We also test the performance of single pruned classification trees (*CART*), support vector machines with radial basis function (*svm*) and a k-nearest-neighbors classifier (*knn*). For RF, the optimal number of trees is found in an internal cross-validation of the classification error testing $10 * 2^i$ trees with $i = 1 \dots 6$. Trees are grown to full depth. For CART, model complexity is optimized in a ten-fold cross-validation. For the support vector machine (*svm*) the kernel width is obtained in a three-fold cross-validation testing the quartiles of $|x - x'|^2$ [7] and regularization ("cost") is found in another 3-fold cross-validation evaluating the range of $\lambda = 10^{-5 \dots 5}$. A binomial model is modeled on top of the binary decisions [31]. Boosting is "discrete" adaboost, with exponential loss, and in conjunction with a bagging of the samples. We test the performance for $10 * 1 \dots 5$ iterations in a 3-fold internal cross-validation. The knn classifier is tested in a threefold cross-validation testing 2^i neighbors with $i = 0 \dots log_2(P)$.

Processing Time. Processing time is critical for the generation of large ensembles of classifiers. Using an implementation in native GNU R it took longer to train an oRF than training an RF (using a Fortran implementation), or the SVM (C); it was still faster than adaboost (C). Growing 100 oblique trees took about 10-100s for most of the learning tasks (on a standard personal computer) with *lda* at the lower end of this time range and *ridge* – requiring parameter tuning – at the upper. Training the node models on a subsampled set of observations, as

done in computer vision applications [9,45], may reduce the overall computation time significantly for large data sets.

Choice of oRF Default Parameters. Random forests are relatively insensitive to the choice of model parameters, and we want to apply all oRF with the same default parameters. We test the effect of forest size, n_{tree}, on the overall classification performance of the classifiers (as determined in a ten-fold cross-validation; Fig. 3, left). Overtraining by increasing n_{tree} could hardly be provoked for any random forest on any data set. Little improvement in classification accuracy is observed for $n_{tree} > 100$. We chose to set $n_{tree} = 300$ for all further experiments. The second tuning parameter is m_{try} – the number of features which are randomly sampled at each node. A popular default choice for RF is $m_{try}^{def} = \sqrt{P}$, with P being the number of features [20]. We find this default to perform reasonably well (Fig. 3, right), although larger differences can be observed for spectral data sets. In all further experiments we use $m_{try} = \sqrt{P}$.

Evaluation. We apply all classifiers to the data set in ten repeats of a ten-fold cross-validation (Table 1) and evaluate both the mean accuracy and the receiver-operator-characteristics with its area-under-the-curve (ROC AUC). ROCs are calculated on the complete test data, and individually on each of the ten resampled data sets, to have a distribution of ten different ROCs which is used to measure significance of differences. To this end we use nonparametric statistical tests. We first identify the best method for a particular data set (defined by the highest mean classification accuracy, underlined in Table 1), and then test whether other classification results differed significantly from this one "best" result. We use paired Cox-Wilcoxon tests at 5% level (on the 100 test sets for classification accuracy and 10 repeats for AUC ROC) to compare the performance between the "best" classifier, and any other classifier applied to the same data set [18]. Classification results which do *not* differ significantly from the approach with the highest accuracy, are indicated by bold font in Table 1.

We analyze the general behavior of the algorithms over data which share certain properties. Trees with oblique and orthogonal perform differently on data with factorial and numerical features [8]. Random forest perform well in applications with many irrelevant predictors and few observations [23]. So, we group the data into three classes (Table 1): data with factorial or discrete features ("factorial data", data sets 1-10 in Table 1), data with continuous numerical features ("numerical data",11-20), and data with many correlated numerical features and few samples ("spectral data", 21-40).

Results are shown in Table 1, the frequency how often a particular method performed best or comparable to the best within the three data classes is reported in Table 2. The optimal choice for factorial data are methods evaluating univariate splits – adaboost, RF and in some classification tasks even CART performed best. They rank – in terms of classification performance – in front of all other methods, with adaboost being first. The advantage on factorial features may be expected as univariate trees consider the relative ordering of observations only. They are insensitive to the arbitrary assignment of numerical values

Table 1. Classification accuracy (left block) and 1−ROC AUC (right block) of the classifiers on the 40 data sets. Data sets are grouped in factorial, nominal and spectral data. Underlined is the best results, bold results do not differ significantly from the best.

		svm	knn	CART	adaboost	RF	RF-rnd	oRF-rnd	oRF-lda	oRF-ridge	svm	knn	CART	adaboost	RF	RF-rnd	oRF-rnd	oRF-lda	oRF-ridge
1	chess	2.8	5.4	2.4	**0.6**	1.4	52.2	23.5	4.1	2.7	0.3	1.2	0.9	**0**	0.1	33.3	0.9	0.4	0.3
2	credit	15.3	32	14.8	**12.9**	**12.7**	17.9	15.4	**13**	44.5	10.3	27.3	15.4	**6.7**	**6.8**	7.5	8.3	**7**	53.3
3	heart	23	35.5	21.3	**18.3**	**17.8**	40.6	20	**18**	**17.9**	15.7	30.9	20.5	11.4	10.4	13.5	12.7	**9.9**	**9.9**
4	hepatitis	20.9	21.6	**20.3**	**24.4**	**17.8**	**18.7**	**19.1**	**19.3**	20.6	47	42.2	28.7	18.9	**14.6**	19.5	16.2	15.7	50
5	monks3	3.6	**1.6**	**1.1**	**1.2**	2.4	43.3	5.7	3.5	**1.3**	1.4	**0.9**	1.7	**0.8**	1.1	5.8	**0.8**	1.1	1
6	promotergene	17.1	20.3	30.4	**9.6**	**11**	50.1	20.8	16.9	18.5	8.8	12	31.3	**3.9**	**4**	32	12.8	7.3	7.4
7	tic-tac-toe	12	16.1	26.5	**0**	4.7	65.3	12.7	15.2	14.8	4	32	31.1	**0**	0.3	73.3	4.9	3.5	2.8
8	titanic	**21.6**	23.3	23	**21.3**	22.7	30.2	26.8	22	24.7	20.1	20.1	25.5	**16.1**	16.8	22.1	20.3	17.6	18.6
9	votes	9.3	12.5	**4.4**	**4.5**	**4.2**	7.7	5.7	5.6	5.1	4.2	8.7	6.1	1.4	**0.9**	1.8	1.4	1.4	1.3
10	cancer	4.3	**3.4**	6.2	3.7	**3.3**	**3**	**2.8**	**3**	**3**	2.3	1.3	4.8	1	0.9	0.9	**0.7**	0.9	0.9
11	ionosphere	**5.1**	14	10.5	**6.4**	**6.5**	7.5	**5.6**	**5.4**	**5.6**	2	16.9	12.3	3.3	2.1	2.1	**1.7**	**1.7**	**1.6**
12	sonar	17.5	18.2	27.2	**13.1**	**16.1**	16.3	**14.4**	18	18.2	7.6	17.3	25.7	**6.1**	6.8	**6.4**	**5.8**	6.5	**6.2**
13	musk	10.4	14.6	19.6	**8.8**	10.6	17.9	10.2	10.6	10.4	3.8	12.9	15.9	**2.9**	4.4	5.3	3.7	5	4.5
14	liver	30.9	32	34.1	**26.8**	**26.8**	**27.8**	27.8	**25.8**	**26.2**	26.6	31.3	38.2	22.3	23.3	23.5	23.4	**21.4**	**21.3**
15	diabetes	30.7	31.7	34.2	**26.6**	**26.7**	**27**	**27.7**	**26.1**	**26.2**	26.8	30	37.7	22.3	23.3	23.3	23.1	**21.2**	**21.5**
16	ringnorm	**12.4**	17	19.8	13.2	17.5	17.9	18	18	18	**2.1**	25.5	40.9	7.2	24.7	14.2	17.9	36.3	35.4
17	spirals	**5.2**	**5.1**	10.3	6.1	6	6.1	**5.5**	**5.3**	**5.2**	**1.1**	1.5	6.9	2.2	2.1	2.1	1.3	1.4	1.3
18	threenorm	**13.3**	14.9	33.8	15.3	16.4	15.1	**14**	14.6	14.7	**6**	6.6	30.6	7.7	7.8	7.4	6.7	7.2	7
19	twonorm	**2.2**	**1.9**	21.6	3.4	3.5	2.8	**2.2**	**1.8**	**1.7**	0.2	0.2	18.5	0.4	0.5	0.3	0.2	**0.2**	**0.2**
20	circles	**2.1**	2.5	5.1	**2.4**	2.8	3.3	**1.7**	**1.9**	**1.8**	0.1	1.8	3.1	0.3	0.4	0.4	**0**	**0**	**0**
21	digits 2-4	1.3	**0.4**	5	2.4	1.7	50	1.5	2.9	2.7	**0.1**	0.3	3.4	0.3	0.1	5.9	0.1	0.5	0.6
22	digits 3-8	**2.3**	3.6	6	**3.1**	**3.2**	50	4.3	4.9	5.2	**0.3**	2.3	4.4	0.6	0.5	30.7	0.8	1.2	1.3
23	digits even-odd	**7.4**	**7.2**	15.8	9.1	**7.8**	50	9.8	16.2	16.6	**2.4**	6.2	14.6	3.2	**2.3**	47.2	3.3	9	9.4
24	RS SRTM	2.5	3.2	3	**0.8**	**0.7**	1.7	1.7	**0.7**	**1**	0.3	2.7	2.7	0.1	**0**	0.1	0.1	**0**	**0**
25	RS ASTER	25.5	29.8	32.7	22.9	24.9	27.5	25.1	**19.3**	**19.9**	18.7	25.4	30.1	17.4	19.6	20.8	18.6	**13.2**	14
26	MRS quality	**6.2**	6.7	18.1	6.7	7.7	9.3	8.4	**6.2**	**6.2**	2.1	2.5	15.5	2.4	2.1	2.7	2.4	1.9	**1.9**
27	MRS tumor 1	**11.2**	12	17.5	**10.6**	**10.4**	12.2	**10.7**	**10.6**	**11**	4.9	8.4	16.8	6.2	**4.6**	5.9	4.9	**4.6**	**4.7**
28	MRS tumor 1	**19.1**	**19.7**	22.3	**18.9**	**19**	**19.1**	**19.1**	**19.7**	**20.1**	28	29.5	35.4	**26.8**	**24.7**	**26**	25.9	26.9	**25.5**
29	IR BSE 1	22.5	23.1	25.2	22.1	20.5	23	23	**13.1**	**13.4**	22	33.6	39.3	27.4	21.3	26.8	24.5	**9.2**	9.9
30	IR BSE 2	27.9	42.9	24.1	25.9	25.7	29.6	34.5	**15**	**15.5**	20.5	41.5	30	20.7	18.5	25.1	27.3	**6.9**	**7.1**
31	IR BSE 3	27.1	40.5	25.2	33.5	24.6	30.8	35.4	**14.9**	**12.6**	20.3	39	31.5	18.2	18.5	26.1	29	**5**	**5.2**
32	MRS yeast 1	**4.3**	9	14.5	7.8	7.3	8.9	7.9	**4.1**	**4.3**	**1.2**	6.4	15.5	3.4	2.9	3.6	2.9	**1.2**	**1.2**
33	MRS yeast 2	**2.4**	**3.2**	9	8.3	3.9	4.4	**3.5**	**3**	**2.8**	**2.2**	4.3	11.1	4.5	3.6	2.8	2.9	2.7	2.8
34	MRS yeast 3	**3**	4.9	8.7	7.2	5	5.2	4.4	**3.2**	**3.2**	3.9	4.8	13.6	9.5	4.8	5	4.8	**3.2**	**3.2**
35	MRS yeast 4	9.7	11	15.7	15.8	12	14.1	13.3	**6.5**	**5.9**	**4.2**	14.2	26.9	6.1	8.6	8	7.1	4.5	**4.2**
36	MRS yeast 5	5	7.1	8	7.4	6.4	6.4	6.4	**3.5**	**3.9**	**3.2**	8.6	16.2	9.9	4.3	5.6	5.5	**3.4**	**3.1**
37	IR wine origin 1	**27.2**	40.7	**26**	**22.1**	**21.7**	**26.4**	29.6	**21.4**	**21.6**	23.5	40.6	28	16.9	**14.5**	19.6	23.8	**13.6**	**13.2**
38	IR wine origin 2	**25.5**	40.6	30.3	**21.8**	**21.1**	**25.4**	32.1	**25.5**	**22.6**	23.2	41.7	31.9	27.2	**15**	19.2	27.5	15.6	**13.9**
39	IR wine grape 1	17.1	40.3	14.7	18	**11.1**	21.9	25.1	**8.4**	**4.6**	6	38.4	18.8	7.6	3	10.2	16.7	**0**	**0**
40	IR wine grape 2	18	38.2	**15.3**	**12.5**	**10.3**	22.1	29.5	**11.6**	**11.1**	6.2	35.2	18.3	5.5	2.7	10.1	21.2	**0**	**0**

Table 2. Ranking of the classifiers summarizing performance for the three different data classes. The figure shows how often a classifier performed best or similar to the best in Table 1. With the exception of of factorial data oblique random forests outperform the alternative classifiers. Overall oRF-ridge performs best.

Ranking		1		2		3		4		5	
Factorial Data	AUC ROC	adaboost	(6)	RF	(4)	oRF-lda	(2)	oRF-rnd	(2)	knn	(1)
	Accuracy	adaboost	(9)	RF	(6)	oRF-lda	(4)	oRF-ridge	(3)	CART	(3)
Nominal Data	AUC ROC	oRF-ridge	(6)	oRF-lda	(5)	svm	(3)	oRF-rnd	(3)	adaboost	(2)
	Accuracy	oRF-rnd	(7)	oRF-ridge	(6)	oRF-lda	(6)	adaboost	(6)	svm	(6)
Spectral Data	AUC ROC	oRF-ridge	(14)	oRF-lda	(12)	svm	(7)	RF	(6)	adaboost	(1)
	Accuracy	oRF-ridge	(17)	oRF-lda	(17)	svm	(10)	RF	(9)	adaboost	(7)

to different factors. On numerical data oblique random forests perform slightly better than alternative classifiers (e.g., ada, SVM). Those oRF with regularized node model perform slightly better than the other two in terms of class separation (i.e, ROC). On spectral data, oRF is a clear first, both in terms of accuracy and ROC. SVM, RF or adaboost perform well on few data sets (*MRS tumor*, *IR yeast 2*) and here most oRF perform equally well. An advantage of oRF-ridge over oRF-lda becomes apparent when comparing class separation. The somewhat weaker general performance of the unregularized oRF suggest that the overall performance may benefit to a large extend from the optimal choice of λ and a sufficiently strong regularization.

Overall, we find that random forests with orthogonal trees (RF) perform well on factorial data, but they are outperformed by oblique random forests (oRF) on numerical and spectral data. Oblique random forests with regularized node models (oRF-ridge) rank overall first, followed by oRF with unregularized model (oRF-lda), both outperforming Breiman's oRF with random split model (oRF-rnd). So, in any learning task which do not comprise discrete factorial features, the oRF may be advantageous over the regular RF and a better choice when an "out-of-the-box" learning algorithm is required.

4 Advantages of Oblique Model Trees

In another set of experiments we want to shed light on specific advantages of the oRF classifier and properties of those types of data it performs well with.

Topology of the Decision Boundary. To understand why RF and oRF behave differently, we visualize the actual class separation for the "mixture" data set [16] (Fig. 4). When testing a single pruned orthogonal tree (CART) we obtain a clear, binary separation of the feature space. However, even when pooling many trees the boundary imposed by the RF keep their blocked, or 'stair-like' topology. They extrapolate the decision boundary in space with few samples through axis-parallel, orthogonal splits. In addition to the blocked decision boundary, this somewhat arbitrary extrapolation may not be a natural choice in high dimensional tasks with few samples and correlated feature, e.g., on spectral data. The oRF adapts here closely to the training data (dots) and to the true boundary of the underlying distribution (dashed lines). In this the separation of the feature space imposed by the oRF is more similar to the SVM (not shown here) than to the RF, both favoring smooth decision boundaries. While for the RBF-kernel SVM the smoothness of this boundary is a parameter to be optimized during training, the random forest framework does not require to adjust any such parameter.

Advantage over Univariate and Random Multivariate Splits. One may argue that RF suffers from significant bias imposed through the topological constraints of the decision boundaries of its base learners. We calculate bias and variance for results from Table 1 using ensemble votes as for regression tasks [16].

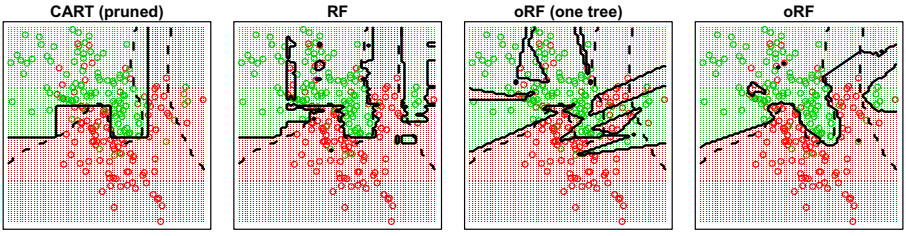


Fig. 4. Visualization of decision boundaries using the mixtures of Gaussians example from [16]. Colors indicate class assignments of the bivariate grid points when training the classifier using the indicated samples (circles). The solid black line represents the decision boundary learned. The dashed black line is Bayes optimal border between the distributions generating the training samples. Shown are single univariate classification tree (CART) and RF, and a single oblique tree and oRF. Differences can be observed in the topology of the decision boundary, and the way decision boundaries are extrapolated in regions with few observations. Note that CART has been optimized using cross-validation, while the others are applied out of the box.

Fig. 5 (left) reports differences in bias and variance for RF and oRF, pooled over the different data types. The advantage of RF on factorial data is due to a lower bias. The higher accuracy of oRF on spectral and numerical data correlates with a lower bias, too. Variance is slightly lower for oRF. A similar analysis can be done to compare the oRF with learned node model – proposed in this study – with the original oRF with random split (Fig. 5, right). Results show that the latter (oRF-rnd) suffers from high bias when compared to the first (oRF-ridge). This suggest that the higher variability of the oRF-rnd, and the resulting lower variability of the ensemble, does not trade off the disadvantage (or bias) of using random split directions.

Advantage on Spectral Data. Oblique random forests perform well on spectral data which shows a strong correlation among the features (Fig. 1). To study this in a controlled fashion, we systematically increase correlation between features for selected data sets (Fig. 6) by adding a random scalar to the features

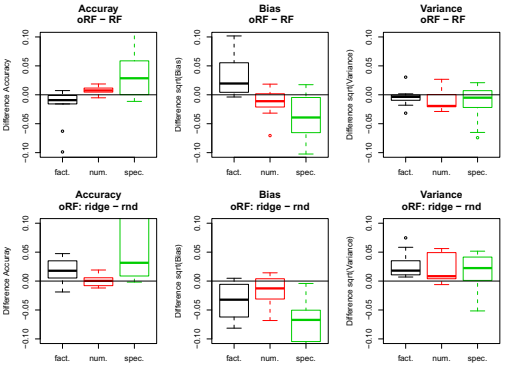


Fig. 5. Comparison of bias and variance between RF and oRF-ridge (left) and oRF-rnd and oRF-ridge (right) pooled over the data types. Boxplots indicate quartiles and outliers. Advantages in classification accuracy are dominated by bias. This benefits oRF-ridge for numerical and spectral data. The bias may arise from the topology of the base learner.

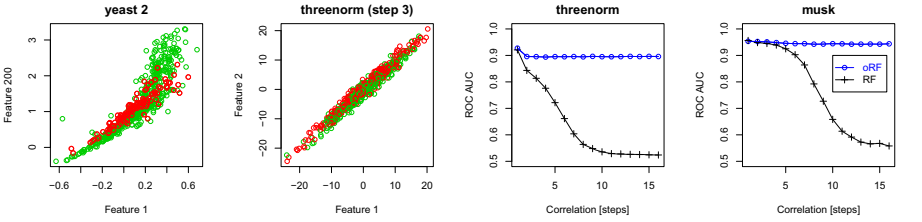


Fig. 6. Correlation between features (left) and performance on artificially correlated data (right). Shown is the 'natural' correlation between features in the *yeast 2* data set and samples of the *threenorm* data set with artificially induced correlation (left, corresponding to 'step 3' in the right figures). While the performance of the oRF remains nearly unchanged even on highly correlated data (right), RF fails beyond a certain point.

of every single observation, drawn from a normal distribution. We then increase the size of this random offset stepwise by a multiplicative factor ($10^{\{0,1,\ldots,15\}}$). As visualized in Fig. 6, this stretches the samples successively along the intersecting line of the feature space. Both oRF and RF are tested on any of the 3*16 resulting data sets. Similar to our general evaluation, we calculate the AUC ROC.

We observe that the performance of the orthogonal random forest decrease at a certain point (Fig. 6, right), finally resulting in complete misclassification. On the *threenorm* data (Figure 6, left), the RF drops from more than 85% classification accuracy rapidly to a close to random prediction. At the same time the performance of the oRF remains practically unchanged. These results suggest, somewhat similar to our reasoning at the beginning, that in classification tasks with collinear samples the marginal distributions of the input variables x_i – i.e., the projections of the data into the subspaces evaluated by the orthogonal random forest in search for an optimal split – may lose their power to separate classes (compare the two marginal distributions in Fig. 1, for example).

5 Feature Importance and Sample Proximity

The random forest framework provides additional tools which help to illustrate decision processes within the random forest ensemble, but which also have a value for exploratory data analysis on their own. This are importance scores reporting the relevance of individual features [44] and sample proximity measures for visualizing similarities between individual samples and for mapping natural groupings within the data [38]. We propose similar tools for the oRF.

Feature Importance. Different feature importance measures exist for the random forest framework. One score is the "Gini importance" [5]. This measure is obtained during training by recording the decrease in Gini impurity for every variable, whenever a variable is selected in a split. Averaging this quantity, individually for each variable, and over all splits in a tree and all trees in the ensemble

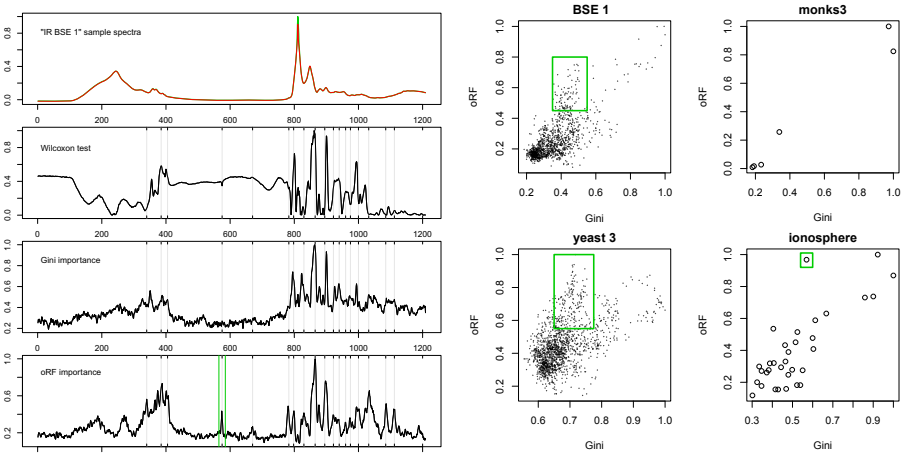


Fig. 7. Random forest feature importance – exemplified for a spectral data sets (BSE 1). The left plot shows different feature important measures, along with a representative spectrum of the data set. For all three tests high values indicate important features. The right plot compares RF and oRF feature importance scores for different data sets. The oRF also assigned importance to a number of additional features which are virtually ignored by the orthogonal RF (green boxes, for BSE 1 data also indicated by green lines).

leads to the Gini importance score. An alternative measure is the "permutation importance". It is obtained by comparing, for each variable, the regular predictions of the classifier with those predictions obtained after randomly permuting the observations for this specific variable. Empirical studies suggest that both the feature rankings – the algorithmically motivated Gini importance, and the statistically defined permutation importance – correlate well in most classification tasks [1]. It has been observed that correlation between features may affect both Gini [23] and permutation importance [28]. For the oRF we obtain a feature relevance measure similar to the Gini importance by calculating analysis of variance (ANOVA) tables at every split in the oblique model tree, and by recording those variables which contribute significantly to the split (as expressed by a sufficiently small p-value, here $p \leq .01$). For every individual variable we record how often it was deemed significant in a split. This frequency, calculated from all splits over a sufficiently large number of trees, is the statistic we propose to use for visualizing feature importance in oblique random forests (Figure 7). We refer to it as "oRF importance" in the following.

We can compare "oRF importance" and "Gini importance". Feature selection is highly relevant in chemometric calibration and classification tasks. So, we show an example for the *BSE 1* data set in Figure 7, a data set where large differences in the performance of RF and oRF can be observed (Table 1). The Gini feature importance is obtained from an ensemble of 1500 trees, and the oRF feature importance using an ensemble with 9000 trees. (For computational convenience we use an oRF with logistic node model.) We chose such high numbers of trees

to guarantee a sufficient sampling of all features with a larger ensemble for the oRF, as oblique model trees need fewer splits than orthogonal decision trees. We also compare both random forest scores with a *t*-test measuring class separation individually at every feature – similar to, for example, fMRI experiments, microarray, or tests on spectra [15]. We observe that both random forest measures show additional peaks when compared with the *t*-test that may be attributed to the presence of patterns which are of multivariate importance [23]. While both Gini and oRF importance show a broad correspondence for most spectral regions, a number of features appear to be relevant to the oRF only (Fig. 7, green boxes and spectra regions). We find such additional "peaks" for most spectral data sets, and some of the numerical data sets (*ionosphere*). It suggests that oblique splits allow the classifier to consider additional spectral regions during classification which are "invisible" to the traditional RF and, consequently, not used during classification.

Sample Proximity. The random forest framework allows one to measure a distance or "proximity" between observations, or samples, in a classification task [5]. During training of the classifier the out-of-bag samples are pushed down the tree. The frequency of how often each pair of samples ends up in the same terminal node – i.e., in the same partition of the feature space – is recorded. After a large number of trees has been learned, and every pair has been tested sufficiently often, these co-occurrence frequencies can be normalized and an affinity matrix is obtained [5,38]. Similar to [5] the affinity matrix can be transformed to a distance matrix counting how often a pair of sample has *not* been in the same node, and the scores from projecting samples to the principal components of this distance matrix can be visualized. It may help to uncover natural groupings in the data and to identify consistently misclassified sample in a very intuitive fashion [38].

Figure 8 shows a projection of the samples to the principal eigen-spaces of their distance matrix for both a RF (lower triangle of the pairs plot matrix) and an oRF (upper triangle). Stark differences are visible: Inter-sample differences in the oRF proximity correlate well with the first eigen-direction of the distance matrix, representing the learned inter-class difference (Fig. 8, right). Higher components show random noise only and no further grouping can be observed. This is very different from the RF proximity, which reveals complex structures even for higher components. This may indicate that orthogonal trees separate feature space by splits which are not required by the learning task, but are imposed by the topology of the base learner.

The observation from the *yeast 2* data set – that oRF show complex inter-sample differences unrelated to the imposed inter-class separation task – holds true, to a lesser extent, for most data sets in our study. It is even visible from the variance explained along the eigen-directions of the affinity matrices. We find for all data sets the variance explained by the first eigen-space of the oRF to be much larger than the variance explained by the first eigen-space of the RF. For higher components the opposite is true indicating that some of the inter-sample differences in the RF are not due to plain inter-class differences imposed

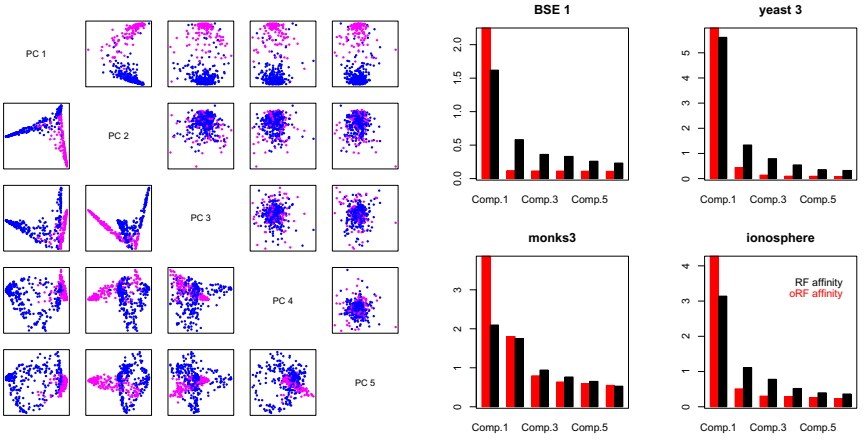


Fig. 8. Visualization of the random forest sample proximity. Shown are sample projected to the principal eigen-spaces of the proximity matrix of RF (left, lower triangle) and oRF (left, upper triangle) and the variation explained by RF (right, black) and oRF (right, red) for further data sets. The oRF is void of structures resulting from the topology of the base learner (*yeast 1*). For oRF, inter-sample differences of the dominating first eigen-direction correlate well with learned inter-class difference. Higher components show random noise and their contribution to inter-sample differences can be neglected here. The RF affinity matrix reveals complex structures even for higher components suggesting that rules by the oRF are much simpler than those induced by the RF. The oRF proximity may be highly preferable for visualizing natural groupings.

by the learning task, but due to a consistently over-complex segmentation of the features space by the splits of the orthogonal trees. Again, this implies that rules by the oRF are much simpler than those induced by the RF and, supposedly, void of complex structures imposed superficially by the topology of the base learner.

6 Conclusion

Random forest with oblique node models have been somewhat neglected since proposed by Breiman in 2001. When replacing the random node model with learned model splits we obtain a classifier that can be implemented straightforwardly, and that performs well on a large range of data sets even with default parameterizations.

– We find random forests with orthogonal splits (RF) to perform well on factorial data. On all other data random forests with oblique node model (oRF) perform better. Here, a learned node model performs better than a random split, and a learned node model with adaptive regularization better than one without. On numerical and spectral data, oRFs outperform a range of alternative classifiers even with default parametrization. The oRF does

exceptionally well in separating high dimensional distributions even when large correlations between features are present.
- The oRF can be used to define feature importance and sample proximity measures. They may be preferable in the analysis of learning tasks where oRF performs exceptionally well and shows less topological bias than RF. These are learning tasks with few samples, many irrelevant features and correlated predictors arising, for example, in biomedical diagnostics [19,39,29,14,25,23]
- The inspection of the feature importance measure suggests that the oRF is able to consider information from variables which are "invisible" to univariate split models (Fig. 1) and void of structures reflecting constraints resulting from the geometry of the base learner. The ability to use these additional variables and considering their contribution to the classification problem illustrates why the oRF perform better than the "traditional" RF on numerical and spectral data.

These results may lead to further work. Oblique decision trees show similarities with deep learning architectures [36], and "reusing" scores from earlier splits for decisions in later nodes would even enhance this similarity. Reusing classification results and probabilities from earlier trees would be very similar to the "auto-context" idea from [42]. One may assume that mixing orthogonal and oblique splits within a tree would help to combine desired properties of both approaches.

References

1. Archer, K.J., Kimes, R.V.: Empirical characterization of random forest variable importance measures. Comput. Stat. Data Anal. 52, 2249–2260 (2008)
2. Biau, G., Devroye, L., Lugosi, G.: Consistency of random forests and other averaging classifiers. J. Mach. Learn. Res., 2015–2033 (2008)
3. Breiman, L.: Bagging predictors. Mach. Learn. 24, 123–140 (1996)
4. Breiman, L.: Arcing classifiers. Technical Report, UC Berkeley (1998)
5. Breiman, L.: Random forests. Mach. Learn. J. 45, 5–32 (2001)
6. Breiman, L.: Consistency for a simple model of random forests. Tech. Rep. 670, UC Berkeley (2004)
7. Caputo, B., Sim, K., Furesjo, F., Smola, A.: Appearance-based object recognition using SVMs: which kernel should I use? In: Proc NIPS WS (2002)
8. Chan, K.Y., Loh, W.Y.: LOTUS: An algorithm for building accurate and comprehensible logistic regression trees. J. Comp. Graph. Stat. 13, 826–852 (2004)
9. Criminisi, A., Shotton, J., Bucciarelli, S.: Decision forests with long-range spatial context for organ localization in ct volumes. In: Proc. MICCAI-PMMIA (2009)
10. Dietterich, T.G.: An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Mach. Learn. 40, 139–157 (2000)
11. Frank, I.E., Friedman, J.H.: A statistical view of some chemometrics regression tools. Technometrics 35, 109–135 (1993)
12. Freund, Y., Shapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Vitányi, P.M.B. (ed.) EuroCOLT 1995. LNCS, vol. 904, Springer, Heidelberg (1995)

13. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. 63, 3–42 (2006)
14. Geurts, P., Fillet, M., de Seny, D., Meuwis, M.A., Malaise, M., Merville, M.P., Wehenkel, L.: Proteomic mass spectra classification using decision tree based ensemble methods. Bioinformatics 21, 313–845 (2005)
15. Hastie, T., Tibshirani, R., Eisen, M., Alizadeh, A., Levy, R., Staudt, L., Chan, W., Botstein, D., Brown, P.: Gene shaving as a method for identifying distinct sets of genes with similar expression patterns. Genome Biol. 1, 1–8 (2000)
16. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, 2nd edn. Springer, Heidelberg (2009)
17. Ho, T.K.: The random subspace method for constructing decision forests. IEEE-T Patt. Anal. Mach. Intell. 20, 832–844 (1998)
18. Hothorn, T., Leisch, F., Zeileis, A., Hornik, K.: The design and analysis of benchmark experiments. Tech. rep., TU Vienna (2003)
19. Jiang, H., Deng, Y., Chen, H.S., Tao, L., Sha, Q., Chen, J., Tsai, C.J., Zhang, S.: Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. BMC Bioinformatics 5(81) (2004)
20. Liaw, A., Wiener, M.: Classification and regression by randomForest. R News 2, 18–22 (2002)
21. Lin, Y., Jeon, Y.: Random forests and adaptive nearest neighbors. J. Am. Stat. Assoc. 101, 578–590 (2006)
22. Martinez-Munoz, G., Hernandez-Lobato, D., Suarez, A.: An analysis of ensemble pruning techniques based on ordered aggregation. IEEE-T Pattern Anal. Mach. Intell. 31, 245–259 (2009)
23. Menze, B.H., Kelm, B.M., Masuch, R., Himmelreich, U., Petrich, W., Hamprecht, F.A.: A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. BMC Bioinformatics 10, 213 (2009)
24. Menze, B.H., Lichy, M.P., Bachert, P., Kelm, B.M., Schlemmer, H.P., Hamprecht, F.A.: Optimal classification of long echo time in vivo magnetic resonance spectra in the detection of recurrent brain tumors. NMR Biomed. 19, 599–610 (2006)
25. Menze, B.H., Petrich, W., Hamprecht, F.A.: Multivariate feature selection and hierarchical classification for infrared spectroscopy: serum-based detection of bovine spongiform encephalopathy. Anal. Bioanal. Chem. 387, 801–1807 (2007)
26. Menze, B.H., Ur, J.A., Sherratt, A.G.: Detection of ancient settlement mounds – Archaeological survey based on the SRTM terrain model. Photogramm Engin. Rem. Sens. 72, 321–327 (2006)
27. Murthy, S.K., Kasif, S., Salzberg, S.: A system for induction of oblique decision trees. J. Artif. Intell. Res. 2, 1–32 (1994)
28. Nicodemus, K., Malley, J., Strobl, C., Ziegler, A.: The behaviour of random forest permutation-based variable importance measures under predictor correlation. BMC Bioinformatics 11, 110 (2010)
29. Pal, M.: Random forest classifier for remote sensing classification. Intern. J. Remote Sensing 1, 217–222 (2005)
30. Pisetta, V., Jouve, P.-E., Zighed, D.A.: Learning with ensembles of randomized trees: New insights. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6323, pp. 67–82. Springer, Heidelberg (2010)
31. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In: Smola, A., Bartlett, P., Schoelkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers. MIT Press, Cambridge (2000)

32. Robnik-Šikonja, M.: Improving random forests. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 359–370. Springer, Heidelberg (2004)
33. Rodriguez, J., Kuncheva, L., Alonso, C.: Rotation forest: A new classifier ensemble method. IEEE T. Patt. Anal. Mach. Intell. 28, 1619–1630 (2006)
34. Saeys, Y., Abeel, T., Van de Peer, Y.: Robust feature selection using ensemble feature selection techniques. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 313–325. Springer, Heidelberg (2008)
35. Segal, M.R.: Machine learning benchmarks and random forest regression. Tech. rep., UC San Francisco (2004)
36. Sethi, I.K.: Entropy nets: from decision trees to neural networks. Proc. IEEE 78, 1605–1613 (1990)
37. Shen, K.Q., Ong, C.J., Li, X.P., Zheng, H., Wilder-Smith, E.P.V.: A feature selection method for multi-level mental fatigue EEG classification. IEEE-T. Biomed. Engin. 54, 1231–1237 (2007) (in press, epub ahead)
38. Su, X., Tsai, C.L., Wang, H., Nickerson, D.M., Li, B.: Subgroup analysis via recursive partitioning. J. Mach. Learn. Res. 10, 141–158 (2009)
39. Svetnik, V., Liaw, A., Tong, C., Culberson, J.C., Sheridan, R.P., Feuston, B.P.: Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. J. Chem. Inf. Model 43, 1947–1958 (2003)
40. Tan, P.J., Dowe, D.L., Webb, G.I., Yu, X.: MML inference of oblique decision trees. In: Proc. AJCAI, pp. 1082–1088 (2004)
41. Tan, P.J., Dowe, D.L.: Decision forests with oblique decision trees. In: Gelbukh, A., Reyes-Garcia, C.A. (eds.) MICAI 2006. LNCS (LNAI), vol. 4293, pp. 593–603. Springer, Heidelberg (2006)
42. Tu, Z., Bai, X.: Auto-context and its application to high-level vision tasks and 3d brain image segmentation. IEEE-T. Patt. Anal. Mach. Intell. 99(preprint) (2009)
43. Tu, Z.: Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. In: Proc. ICCV, pp. 1589–1596 (2005)
44. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature selection with ensembles, artificial variables, and redundancy elimination. J. Mach. Learn. Res. 10, 1341–1366 (2009)
45. Yao, B., Khosla, A., Fei-Fei, L.: Combining randomization and discrimination for fine-grained image categorization. In: Proc. CVPR (2011)

An Alternating Direction Method for Dual MAP LP Relaxation

Ofer Meshi and Amir Globerson

The School of Computer Science and Engineering,
The Hebrew University of Jerusalem, Jerusalem, Israel
{meshi,gamir}@cs.huji.ac.il

Abstract. Maximum a-posteriori (MAP) estimation is an important task in many applications of probabilistic graphical models. Although finding an exact solution is generally intractable, approximations based on linear programming (LP) relaxation often provide good approximate solutions. In this paper we present an algorithm for solving the LP relaxation optimization problem. In order to overcome the lack of strict convexity, we apply an augmented Lagrangian method to the dual LP. The algorithm, based on the alternating direction method of multipliers (ADMM), is guaranteed to converge to the global optimum of the LP relaxation objective. Our experimental results show that this algorithm is competitive with other state-of-the-art algorithms for approximate MAP estimation.

Keywords: Graphical Models, Maximum a-posteriori, Approximate Inference, LP Relaxation, Augmented Lagrangian Methods.

1 Introduction

Graphical models are widely used to describe multivariate statistics for discrete variables, and have found widespread applications in numerous domains. One of the basic inference tasks in such models is to find the *maximum a-posteriori* (MAP) assignment. Unfortunately, this is typically a hard computational problem which cannot be solved exactly for many problems of interest. It has turned out that *linear programming* (LP) relaxations provide effective approximations to the MAP problem in many cases (e.g., see [15, 21, 24]).

Despite the theoretical computational tractability of MAP-LP relaxations, solving them in practice is a challenge for real world problems. Using off-the-shelf LP solvers is typically inadequate for large models since the resulting LPs have too many constraints and variables [29]. This has led researchers to seek optimization algorithms that are tailored to the specific structure of the MAP-LP [7, 13, 14, 16, 20, 28]. The advantage of such methods is that they work with very simple local updates and are therefore easy to implement in the large scale setting.

The suggested algorithms fall into several classes, depending on their approach to the problem. The TRW-S [14], MSD [28] and MPLP [7] algorithms employ

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 470–483, 2011.

coordinate descent in the dual of the LP. While these methods typically show good empirical behavior, they are not guaranteed to reach the global optimum of the LP relaxation. This is a result of non strict-convexity of the dual LP and the fact that block coordinate descent might get stuck in suboptimal points under these conditions. One way to avoid this problem is to use a *soft-max* function which is smooth and strictly convex, hence this results in globally convergent algorithms [6, 10, 12]. Another class of algorithms [13, 16] uses the same dual objective, but employs variants of subgradient descent to it. While these methods are guaranteed to converge globally, they are typically slower in practice than the coordinate descent ones (e.g., see [13] for a comparison). Finally, there are also algorithms that optimize the primal LP directly. One example is the proximal point method of Ravikumar et al. [20]. While also globally convergent, it has the disadvantage of using a double loop scheme where every update involves an iterative algorithm for projecting onto the local polytope.

More recently, Martins et al. [17] proposed a globally convergent algorithm for MAP-LP based on the *alternating direction method of multipliers* (ADMM) [8, 5, 4, 2]. This method proceeds by iteratively updating primal and dual variables in order to find a saddle point of an *augmented Lagrangian* for the problem. They suggest to use an augmented Lagrangian of the *primal* MAP-LP problem. However, their formulation is restricted to binary pairwise factors and several specific global factors. In this work, we propose an algorithm that is based on the same key idea of ADMM, however it stems from augmenting the Lagrangian of the *dual* MAP-LP problem instead. An important advantage of our approach is that the resulting algorithm can be applied to models with *general local factors* (non-pairwise, non-binary). We also show that in practice our algorithm converges much faster than the primal ADMM algorithm and that it compares favorably with other state-of-the-art methods for MAP-LP optimization.

2 MAP and LP Relaxation

Markov Random Fields (MRFs) are probabilistic graphical models that encode the joint distribution of a set of discrete random variables $\mathcal{X} = \{X_1, ..., X_n\}$. The joint probability is defined by combining a set C of local functions $\theta_c(x_c)$, termed *factors*. The factors depend only on (small) subsets of the variables ($X_c \subseteq \mathcal{X}$) and model the direct interactions between them (to simplify notation we drop the variable name in $X_c = x_c$; see [27]). The joint distribution is then given by: $P(x) \propto \exp\left(\sum_i \theta_i(x_i) + \sum_{c \in C} \theta_c(x_c)\right)$, where we have included also singleton factors over individual variables [27]. In many applications of MRFs we are interested in finding the maximum probability assignment (MAP assignment). This yields the optimization problem:

$$\arg\max_x \sum_i \theta_i(x_i) + \sum_{c \in C} \theta_c(x_c)$$

Due to its combinatorial nature, this problem is NP-hard for general graphical models, and tractable only in isolated cases such as tree structured graphs. This has motivated research on approximation algorithms.

One of the most successful approximation schemes has been to use LP relaxations of the MAP problem. In this approach the original combinatorial problem is posed as a LP and then some of the constraints are relaxed to obtain a tractable LP problem that approximates the original one. In our case, the resulting MAP-LP relaxation problem is:

$$\max_{\mu \in L(G)} \sum_i \sum_{x_i} \mu_i(x_i)\theta_i(x_i) + \sum_c \sum_{x_c} \mu_c(x_c)\theta_c(x_c) \tag{1}$$

where μ are auxiliary variables that correspond to (pseudo) marginal distributions, and $L(G)$ is the reduced set of constraints called the *local polytope* [27], defined by:

$$L(G) = \left\{ \mu \geq 0 \left| \begin{array}{ll} \sum_{x_{c\setminus i}} \mu_c(x_{c\setminus i}, x_i) = \mu_i(x_i) & \forall c, i : i \in c, x_i \\ \sum_{x_i} \mu_i(x_i) = 1 & \forall i \end{array} \right. \right\}$$

In this paper we use the dual problem of Eq. (1), which takes the form:

$$\min_{\delta} \sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c: i \in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \sum_{i: i \in c} \delta_{ci}(x_i) \right) \tag{2}$$

where δ are dual variables corresponding to the marginalization constraints in $L(G)$ (see [22, 28, 23]).[1] This formulation offers several advantages. First, it minimizes an upper bound on the true MAP value. Second, it provides an optimality certificate through the duality gap w.r.t. a decoded primal solution [23]. Third, the resulting problem is unconstrained, which facilitates its optimization. Indeed, several algorithms have been proposed for optimizing this dual problem. The two main approaches are block coordinate descent [14, 28, 7] and subgradient descent [16], each with its advantages and disadvantages. In particular, coordinate descent algorithms are typically much faster at minimizing the dual, while the subgradient method is guaranteed to converge to the global optimum (see [23] for in-depth discussion).

Recently, Jojic et al. [13] presented an accelerated dual decomposition algorithm which stems from adding strongly convex smoothing terms to the subproblems in the dual function Eq. (2). Their method achieves a better convergence rate over the standard subgradient method ($O\left(\frac{1}{\epsilon}\right)$ vs. $O\left(\frac{1}{\epsilon^2}\right)$). An alternative approach, that is also globally convergent, has been recently suggested by Martins et al. [17]. Their approach is based on an augmented Lagrangian method, which we next discuss.

3 The Alternating Direction Method of Multipliers

We now briefly review ADMM for convex optimization [8, 5, 4, 2].

[1] An equivalent optimization problem can be derived via a dual decomposition approach [23].

Consider the following optimization problem:

$$\text{minimize } f(x) + g(z) \quad \text{s.t. } Ax = z \tag{3}$$

where f and g are convex functions. The ADMM approach begins by adding the function $\frac{\rho}{2} \|Ax - z\|^2$ to the above objective, where $\rho > 0$ is a penalty parameter. This results in the optimization problem:

$$\text{minimize } f(x) + g(z) + \frac{\rho}{2} \|Ax - z\|^2 \quad \text{s.t. } Ax = z \tag{4}$$

Clearly the above has the same optimum as Eq. (3) since when the constraints $Ax = z$ are satisfied, the added quadratic term equals zero. The Lagrangian of the augmented problem Eq. (4) is given by:

$$\mathcal{L}_\rho(x, z, \nu) = f(x) + g(z) + \nu^\top (Ax - z) + \frac{\rho}{2} \|Ax - z\|^2 \tag{5}$$

where ν is a vector of Lagrange multipliers. The solution to the problem of Eq. (4) is given by $\max_\nu \min_{x,z} \mathcal{L}_\rho(x, z, \nu)$. The ADMM method provides an elegant algorithm for finding this saddle point. The idea is to combine subgradient descent over ν with coordinate descent over the x and z variables. The method applies the following iterations:

$$\begin{aligned} x^{t+1} &= \arg\min_x \mathcal{L}_\rho(x, z^t, \nu^t) \\ z^{t+1} &= \arg\min_z \mathcal{L}_\rho(x^{t+1}, z, \nu^t) \\ \nu^{t+1} &= \nu^t + \rho \left(Ax^{t+1} - z^{t+1}\right) \end{aligned} \tag{6}$$

The algorithm consists of primal and dual updates, where the primal update is executed sequentially, minimizing first over x and then over z. This split retains the decomposition of the objective that has been lost due to the addition of the quadratic term.

The algorithm is run either until the number of iterations exceeds a predefined limit, or until some termination criterion is met. A commonly used such stopping criterion is: $\|Ax - z\|^2 \leq \epsilon$ and $\left\|z^{t+1} - z^t\right\|^2 \leq \epsilon$. These two conditions can serve to bound the suboptimality of the solution.

The ADMM algorithm is guaranteed to converge to the global optimum of Eq. (3) under rather mild conditions [2]. However, in terms of convergence rate, the worst case complexity of ADMM is $O(\frac{1}{\epsilon^2})$. Despite this potential caveat, ADMM has been shown to work well in practice (e.g., [1, 26]). Recently, accelerated variants on the basic alternating direction method have been proposed [9]. These faster algorithms are based on linearization and come with improved convergence rate of $O(\frac{1}{\epsilon})$, achieving the theoretical lower bound for first-order methods [19]. In this paper we focus on the basic ADMM formulation and leave derivation of accelerated variants to future work.

4 The Augmented Dual LP Algorithm

In this section we derive our algorithm by applying ADMM to the dual MAP-LP problem of Eq. (2). The challenge is to design the constraints in a way that facilitates efficient closed-form solutions for all updates.

To this end, we duplicate the dual variables δ and denote the second copy by $\bar{\delta}$. We then introduce additional variables λ_c corresponding to the summation of δ's pertaining to factor c. These agreement constraints are enforced through $\bar{\delta}$, and thus we have a constraint $\delta_{ci}(x_i) = \bar{\delta}_{ci}(x_i)$ for all $c, i : i \in c, x_i$, and $\lambda_c(x_c) = \sum_{i:i\in c} \bar{\delta}_{ci}(x_i)$ for all c, x_c.

Following the ADMM framework, we add quadratic terms and obtain the augmented Lagrangian for the dual MAP-LP problem of Eq. (2):

$$\begin{aligned}
&\mathcal{L}_\rho(\delta, \lambda, \bar{\delta}, \gamma, \mu) = \\
&\sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \lambda_c(x_c) \right) \\
&+ \sum_c \sum_{i:i\in c} \sum_{x_i} \gamma_{ci}(x_i) \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{i:i\in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2 \\
&+ \sum_c \sum_{x_c} \mu_c(x_c) \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right)^2
\end{aligned}$$

To see the relation of this formulation to Eq. (5), notice that (δ, λ) subsume the role of x, $\bar{\delta}$ subsumes the role of z (with $g(z) = 0$), and the multipliers (γ, μ) correspond to ν.

The updates of our algorithm, which stem from Eq. (6), are summarized in Alg. 1 (a detailed derivation appears in Appendix A). In Alg. 1 we define $N(i) = \{c : i \in c\}$, and the subroutine $w = \text{TRIM}(v, d)$ that serves to clip the values in the vector v at some threshold t (i.e., $w_i = \min\{v_i, t\}$) such that the sum of removed parts equals $d > 0$ (i.e., $\sum_i v_i - w_i = d$). This can be carried out efficiently in linear time (in expectation) by partitioning [3].

Notice that all updates can be computed efficiently so the cost of each iteration is similar to that of message passing algorithms like MPLP [7] or MSD [28], and to that of dual decomposition [13, 16]. Furthermore, significant speedup is attained by caching some results for future iterations. In particular, the threshold in the TRIM subroutine (the new maximum) can serve as a good initial guess at the next iteration, especially at later iterations where the change in variable values is quite small. Finally, many of the updates can be executed in parallel. In particular, the δ update can be carried out simultaneously for all variables i, and likewise all factors c can be updated simultaneously in the λ and $\bar{\delta}$ updates. In addition, δ and λ can be optimized independently, since they appear in different parts of the objective. This may result in considerable reduction in runtime when executed on parallel architecture.[2]

[2] In our experiments we used sequential updates.

Algorithm 1. The Augmented Dual LP Algorithm (ADLP)

for $t = 1$ to T **do**

Update δ: for all $i = 1, ..., n$

Set $\bar{\theta}_i = \theta_i + \sum_{c:i\in c}(\bar{\delta}_{ci} - \frac{1}{\rho}\gamma_{ci})$

$\bar{\theta}'_i = \text{TRIM}(\bar{\theta}_i, \frac{|N(i)|}{\rho})$

$q = (\bar{\theta}_i - \bar{\theta}'_i)/|N(i)|$

Update $\delta_{ci} = \bar{\delta}_{ci} - \frac{1}{\rho}\gamma_{ci} - q \quad \forall c : i \in c$

Update λ: for all $c \in C$

Set $\bar{\theta}_c = \theta_c - \sum_{i:i\in c}\bar{\delta}_{ci} + \frac{1}{\rho}\mu_c$

$\bar{\theta}'_c = \text{TRIM}(\bar{\theta}_c, \frac{1}{\rho})$

Update $\lambda_c = \theta_c - \bar{\theta}'_c$

Update $\bar{\delta}$: for all $c \in C, i : i \in c, x_i$

Set $v_{ci}(x_i) = \delta_{ci}(x_i) + \frac{1}{\rho}\gamma_{ci}(x_i) + \sum_{x_{c\setminus i}} \lambda_c(x_{c\setminus i}, x_i) + \frac{1}{\rho}\sum_{x_{c\setminus i}} \mu_c(x_{c\setminus i}, x_i)$

$\bar{v}_c = \frac{1}{1+\sum_{k:k\in c}|X_{c\setminus k}|}\sum_{k:k\in c}|X_{c\setminus k}|\sum_{x_k} v_{ck}(x_k)$

Update $\bar{\delta}_{ci}(x_i) = \frac{1}{1+|X_{c\setminus i}|}\left[v_{ci}(x_i) - \sum_{j:j\in c, j\neq i}|X_{c\setminus\{i,j\}}|\left(\sum_{x_j} v_{cj}(x_j) - \bar{v}_c\right)\right]$

Update the multipliers:

$\gamma_{ci}(x_i) \leftarrow \gamma_{ci}(x_i) + \rho\left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i)\right)$ for all $c \in C, i : i \in c, x_i$

$\mu_c(x_c) \leftarrow \mu_c(x_c) + \rho\left(\lambda_c(x_c) - \sum_{i:i\in c}\bar{\delta}_{ci}(x_i)\right)$ for all $c \in C, x_c$

end for

5 Experimental Results

To evaluate our augmented dual LP (ADLP) algorithm (Alg. 1) we compare it to two other algorithms for finding an approximate MAP solution. The first is MPLP of Globerson and Jaakkola [7], which minimizes the dual LP of Eq. (2) via block coordinate descent steps (cast as message passing). The second is the accelerated dual decomposition (ADD) algorithm of Jojic et al. [13].[3] We conduct experiments on protein design problems from the dataset of Yanover et al. [29]. In these problems we are given a 3D structure and the goal is to find a sequence of amino-acids that is the most stable for that structure. The problems are modeled by singleton and pairwise factors and can be posed as finding a MAP assignment for the given model. This is a demanding setting in which each problem may have hundreds of variables with 100 possible states on average [29, 24].

Figure 1 shows two typical examples of protein design problems. It plots the objective of Eq. (2) (computed using δ variables only) as a function of the execution time for all algorithms. First, in Figure 1 (left) we observe that the coordinate descent algorithm (MPLP) converges faster than the other algorithms,

[3] For both algorithms we used the same `C++` implementation used by Jojic et al. [13], available at `http://ai.stanford.edu/~sgould/svl`. Our own algorithm was implemented as an extension of their package.

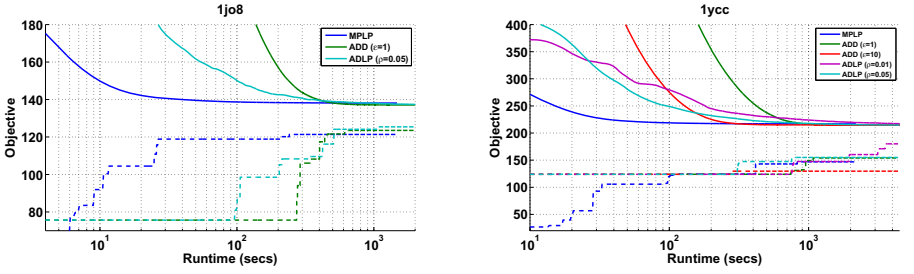


Fig. 1. Comparison of three algorithms for approximate MAP estimation: our augmented dual LP algorithm (ADLP), accelerated dual decomposition algorithm (ADD) by Jojic et al. [13], and the dual coordinate descent MPLP algorithm [7]. The figure shows two examples of protein design problems, for each the dual objective of Eq. (2) is plotted as a function of execution time. Dashed lines denote the value of the best decoded primal solution.

however it tends to stop prematurely and yield suboptimal solutions. In contrast, ADD and ADLP take longer to converge but achieve the globally optimal solution to the approximate objective. Second, it can be seen that the convergence times of ADD and ADLP are very close, with a slight advantage to ADD. The dashed lines in Figure 1 show the value of the decoded primal solution (assignment) [23]. We see that there is generally a correlation between the quality of the dual objective and the decoded primal solution, namely the decoded primal solution improves as the dual solution approaches optimality. Nevertheless, we note that there is no dominant algorithm in terms of decoding (here we show examples where our decoding is superior). In many cases MPLP yields better decoded solutions despite being suboptimal in terms of the dual objective (not shown; this is also noted in [13]).

We also conduct experiments to study the effect of the penalty parameter ρ. Our algorithm is guaranteed to globally converge for all $\rho > 0$, but its choice affects the actual rate of convergence. In Figure 1 (right) we compare two values of the penalty parameter $\rho = 0.01$ and $\rho = 0.05$. It shows that setting $\rho = 0.01$ results in somewhat slower convergence to the optimum, however in this case the final primal solution (dashed line) is better than that of the other algorithms. In practice, in order to choose an appropriate ρ, one can run a few iterations of ADLP with several values and see which one achieves the best objective [17]. We mention in passing that ADD employs an accuracy parameter ϵ which determines the desired suboptimality of the final solution [13]. Setting ϵ to a large value results in faster convergence to a lower accuracy solution. On the one hand, this trade-off can be viewed as a merit of ADD, which allows to obtain coarser approximations at reduced cost. On the other hand, an advantage of our method is that the choice of penalty ρ affects only the rate of convergence and does not impose additional reduction in solution accuracy over that of the LP relaxation. In Figure 1 (left) we use $\epsilon = 1$, as in Jojic et al., while in Figure 1 (right) we compare two values $\epsilon = 1$ and $\epsilon = 10$ to demonstrate the effect of this accuracy parameter.

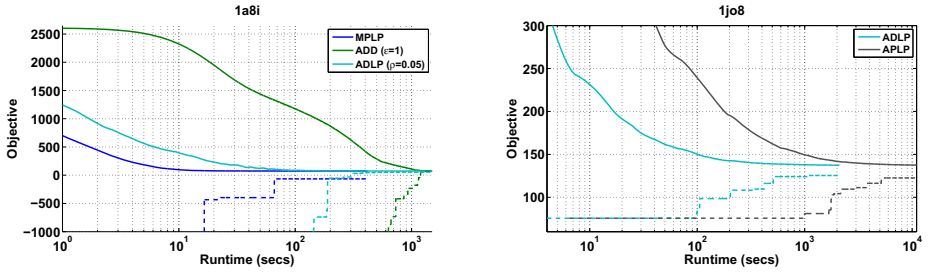


Fig. 2. (Left) Comparison for a side chain prediction problem similar to Figure 1 (left). (Right) Comparison of our augmented dual LP algorithm (ADLP) and a generalized variant (APLP) of the ADMM algorithm by Martins et al. [17] on a protein design problem. The dual objective of Eq. (2) is plotted as a function of execution time. Dashed lines denote the value of the best decoded primal solution.

We next compare performance of the algorithms on a side chain prediction problem [29]. This problem is the inverse of the protein design problem, and involves finding the 3D configuration of rotamers given the backbone structure of a protein. Figure 2 (left) shows a comparison of MPLP, ADD and ADLP on one of the largest proteins in the dataset (812 variables with 12 states on average). As in the protein design problems, MPLP converges fast to a suboptimal solution. We observe that here ADLP converges somewhat faster than ADD, possibly because the smaller state space results in faster ADLP updates.

As noted earlier, Martins et al. [17] recently presented an approach that applies ADMM to the primal LP (i.e., Eq. (1)). Although their method is limited to binary pairwise factors (and several global factors), it can be modified to handle non-binary higher-order factors, as the derivation in Appendix B shows. We denote this variant by APLP. As in ADLP, in the APLP algorithm all updates are computed analytically and executed efficiently. Figure 2 (right) shows a comparison of ADLP and APLP on a protein design problem. It illustrates that ADLP converges significantly faster than APLP (similar results, not shown here, are obtained for the other proteins).

6 Discussion

Approximate MAP inference methods based on LP relaxation have drawn much attention lately due to their practical success and attractive properties. In this paper we presented a novel globally convergent algorithm for approximate MAP estimation via LP relaxation. Our algorithm is based on the augmented Lagrangian method for convex optimization, which overcomes the lack of strict convexity by adding a quadratic term to smooth the objective. Importantly, our algorithm proceeds by applying simple to implement closed-form updates, and it is highly scalable and parallelizable. We have shown empirically that our algorithm compares favorably with other state-of-the-art algorithms for approximate MAP estimation in terms of accuracy and convergence time.

Several existing globally convergent algorithms for MAP-LP relaxation rely on adding local entropy terms in order to smooth the objective [6, 10, 12, 13]. Those methods must specify a temperature control parameter which affects the quality of the solution. Specifically, solving the optimization subproblems at high temperature reduces solution accuracy, while solving them at low temperature might raise numerical issues. In contrast, our algorithm is quite insensitive to the choice of such control parameters. In fact, the penalty parameter ρ affects the rate of convergence but not the accuracy or numerical stability of the algorithm. Moreover, despite lack of fast convergence rate guarantees, in practice the algorithm has similar or better convergence times compared to other globally convergent methods in various settings. Note that [17] also show an advantage of their primal based ADMM method over several baselines.

Several improvements over our basic algorithm can be considered. One such improvement is to use smart initialization of the variables. For example, since MPLP achieves larger decrease in objective at early iterations, it is possible to run it for a limited number of steps and then take the resulting variables δ for the initialization of ADLP. Notice, however, that for this scheme to work well, the Lagrange multipliers γ and μ should be also initialized accordingly. Another potential improvement is to use an adaptive penalty parameter ρ_t (e.g., [11]). This may improve convergence in practice, as well as reduce sensitivity to the initial choice of ρ. On the downside, the theoretical convergence guarantees of ADMM no longer hold in this case. Martins et al. [17] show that the ADMM framework is also suitable for handling certain types of global factors, which include a large number of variables in their scope (e.g., XOR factor). Using an appropriate formulation, it is possible to incorporate such factors in our dual LP framework as well.[4] Finally, it is likely that our method can be further improved by using recently introduced accelerated variants of ADMM [9]. Since these variants achieve asymptotically better convergence rate, the application of such methods to MAP-LP similar to the one we presented here will likely result in faster algorithms for approximate MAP estimation.

In this paper, we assumed that the model parameters were given. However, in many cases one wishes to learn these from data, for example by minimizing a prediction loss (e.g., hinge loss [25]). We have recently shown how to incorporate dual relaxation algorithms into such learning problems [18]. It will be interesting to apply our ADMM approach in this setting to yield an efficient learning algorithm for structured prediction problems.

Acknowledgments. We thank Ami Wiesel and Elad Eban for useful discussions and comments on this manuscript. We thank Stephen Gould for his SVL code. Ofer Meshi is a recipient of the Google European Fellowship in Machine Learning, and this research is supported in part by this Google Fellowship.

[4] The auxiliary variables λ_c are not used in this case.

A Derivation of Augmented Dual LP Algorithm

In this section we derive the ADMM updates for the augmented Lagrangian of the dual MAP-LP which we restate here for convenience:

$$\begin{aligned}
&\mathcal{L}_\rho(\delta, \lambda, \bar{\delta}, \gamma, \mu) = \\
&\sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c: i \in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \lambda_c(x_c) \right) \\
&+ \sum_c \sum_{i: i \in c} \sum_{x_i} \gamma_{ci}(x_i) \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{i: i \in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2 \\
&+ \sum_c \sum_{x_c} \mu_c(x_c) \left(\lambda_c(x_c) - \sum_{i: i \in c} \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i: i \in c} \bar{\delta}_{ci}(x_i) \right)^2
\end{aligned}$$

Updates

- **The δ update:**
  For each variable $i = 1, ..., n$ consider a block δ_i which consists of δ_{ci} for all $c : i \in c$. For this block we need to minimize the following function:

$$\max_{x_i} \left(\theta_i(x_i) + \sum_{c: i \in c} \delta_{ci}(x_i) \right) + \sum_{c: i \in c} \sum_{x_i} \gamma_{ci}(x_i) \delta_{ci}(x_i) + \frac{\rho}{2} \sum_{c: i \in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2$$

  Equivalently, this can be written more compactly in vector notation as:

$$\min_{\delta_i} \frac{1}{2} \|\delta_i\|^2 - (\bar{\delta}_i - \frac{1}{\rho} \gamma_i)^\top \delta_i + \frac{1}{\rho} \max_{x_i} (\theta_i(x_i) + \sum_{c: i \in c} \delta_{ci}(x_i))$$

  where $\bar{\delta}_i$ and γ_i are defined analogous to δ_i. The closed-form solution to this QP is given by the update in Alg. 1. It is obtained by inspecting KKT conditions and exploiting the structure of the summation inside the max (for a similar derivation see [3]).
- **The λ update:**
  For each factor $c \in C$ we seek to minimize the function:

$$\max_{x_c} \left(\theta_c(x_c) - \lambda_c(x_c) \right) + \sum_{x_c} \mu_c(x_c) \lambda_c(x_c) + \frac{\rho}{2} \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i: i \in c} \bar{\delta}_{ci}(x_i) \right)^2$$

  In equivalent vector notation we have the problem:

$$\min_{\lambda_c} \frac{1}{2} \|\lambda_c\|^2 - \left(\sum_{i: i \in c} \bar{\delta}_{ci} - \frac{1}{\rho} \mu_c \right)^\top \lambda_c + \frac{1}{\rho} \max_{x_c} (\theta_c(x_c) - \lambda_c(x_c))$$

  This QP is very similar to that of the δ update and can be solved using the same technique. The resulting closed-form update is given in Alg. 1.

– **The $\bar{\delta}$ update:**
For each $c \in C$ we consider a block which consists of $\bar{\delta}_{ci}$ for all $i : i \in c$. We seek a minimizer of the function:

$$-\sum_{i:i\in c}\sum_{x_i}\gamma_{ci}(x_i)\bar{\delta}_{ci}(x_i) + \frac{\rho}{2}\sum_{i:i\in c}\sum_{x_i}\left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i)\right)^2$$
$$-\sum_{x_c}\mu_c(x_c)\sum_{i:i\in c}\bar{\delta}_{ci}(x_i) + \frac{\rho}{2}\sum_{x_c}\left(\lambda_c(x_c) - \sum_{i:i\in c}\bar{\delta}_{ci}(x_i)\right)^2$$

Taking partial derivative w.r.t. $\bar{\delta}_{ci}(x_i)$ and setting to 0 yields:

$$\bar{\delta}_{ci}(x_i) = \frac{1}{1+|X_{c\setminus i}|}\left(v_{ci}(x_i) - \sum_{j:j\in c, j\neq i}|X_{c\setminus\{i,j\}}|\sum_{x_j}\bar{\delta}_{cj}(x_j)\right)$$

where: $v_{ci}(x_i) = \delta_{ci}(x_i) + \frac{1}{\rho}\gamma_{ci}(x_i) + \sum_{x_{c\setminus i}}\lambda_c(x_{c\setminus i}, x_i) + \frac{1}{\rho}\sum_{x_{c\setminus i}}\mu_c(x_{c\setminus i}, x_i)$. Summing this over x_i and $i : i \in c$ and plugging back in, we get the update in Alg. 1.

– Finally, the multipliers update is straightforward.

B Derivation of Augmented Primal LP Algorithm

We next derive the algorithm for optimizing Eq. (1) with general local factors.

Consider the following formulation which is equivalent to the primal MAP-LP problem of Eq. (1). Define:

$$f_i(\mu_i) = \begin{cases} \sum_{x_i}\mu_i(x_i)\theta_i(x_i) & \mu_i(x_i) \geq 0 \text{ and } \sum_{x_i}\mu_i(x_i) = 1 \\ -\infty & \text{otherwise} \end{cases}$$
$$f_c(\mu_c) = \begin{cases} \sum_{x_c}\mu_c(x_c)\theta_c(x_c) & \mu_c(x_c) \geq 0 \text{ and } \sum_{x_c}\mu_c(x_c) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

f accounts for the non-negativity and normalization constraints in $L(G)$. We add the marginalization constraints via copies of μ_c for each $i \in c$, denoted by $\bar{\mu}_{ci}$. Thus we get the augmented Lagrangian:

$$\mathcal{L}_\rho(\mu, \bar{\mu}, \delta, \beta) =$$
$$\sum_i f_i(\mu_i) + \sum_c f_c(\mu_c)$$
$$-\sum_c\sum_{i:i\in c}\sum_{x_i}\delta_{ci}(x_i)\left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i)\right) - \frac{\rho}{2}\sum_c\sum_{i:i\in c}\sum_{x_i}\left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i)\right)^2$$
$$-\sum_c\sum_{i:i\in c}\sum_{x_c}\beta_{ci}(x_c)\left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c)\right) - \frac{\rho}{2}\sum_c\sum_{i:i\in c}\sum_{x_c}\left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c)\right)^2$$

where $\bar{\mu}_{ci}(x_i) = \sum_{x_{c\setminus i}} \bar{\mu}_{ci}(x_{c\setminus i}, x_i)$.

To draw the connection with Eq. (5), in this formulation μ subsumes the role of x, $\bar{\mu}$ subsumes the role of z (with $g(z) = 0$), and the multipliers (δ, β) correspond to ν. We next show the updates which result from applying Eq. (6) to this formulation.

- **Update** μ_i for all $i = 1, ..., n$:

$$\mu_i \leftarrow \arg\max_{\mu_i \in \Delta_i} \mu_i^\top \left(\theta_i + \sum_{c: i \in c} (\delta_{ci} + \rho M_i \bar{\mu}_{ci}) \right) - \frac{1}{2} \mu_i^\top (\rho |N(i)| I) \mu_i$$

  where $M_i \bar{\mu}_{ci} = \sum_{x_{c\setminus i}} \bar{\mu}_{ci}(x_{c\setminus i}, \cdot)$.
  We have to maximize this QP under simplex constraints on μ_i. Notice that the objective matrix is diagonal, so this can be solved in closed form by shifting the target vector and then truncating at 0 such that the sum of positive elements equals 1 (see [3]). The solution can be computed in linear time (in expectation) by partitioning [3].
- **Update** μ_c for all $c \in C$:

$$\mu_c \leftarrow \arg\max_{\mu_c \in \Delta_c} \mu_c^\top \left(\theta_c + \sum_{i: i \in c} (\beta_{ci} + \rho \bar{\mu}_{ci}) \right) - \frac{1}{2} \mu_c^\top (\rho |N(c)| I) \mu_c$$

  where $N(c) = \{i : i \in c\}$.
  Again we have a projection onto the simplex with diagonal objective matrix, which can be done efficiently.
- **Update** $\bar{\mu}_{ci}$ for all $c \in C, i : i \in c$:

$$\bar{\mu}_{ci} \leftarrow \arg\max_{\bar{\mu}_{ci}} \bar{\mu}_{ci}^\top \left(M_i^\top (\rho \mu_i - \delta_{ci}) - \beta_{ci} + \rho \mu_c \right) - \frac{\rho}{2} \bar{\mu}_{ci}^\top \left(M_i^\top M_i + I \right) \bar{\mu}_{ci}$$

  Here we have an unconstrained QP, so the solution is obtained by $H^{-1}v$. Further notice that the inverse H^{-1} can be computed in closed form. To see how, $M_i^\top M_i$ is a block-diagonal matrix with blocks of ones with size $|X_i|$. Therefore, $H = \rho \left(M_i^\top M_i + I \right)$ is also block-diagonal. It follows that the inverse H^{-1} is a block-diagonal matrix where each block is the inverse of the corresponding block in H. Finally, it is easy to verify that the inverse of a block $\rho \left(1_{|X_i|} + I_{|X_i|} \right)$ is given by $\frac{1}{\rho} \left(I_{|X_i|} - \frac{1}{|X_i|+1} 1_{|X_i|} \right)$.
- **Update the multipliers**:

$$\begin{aligned} \delta_{ci}(x_i) &\leftarrow \delta_{ci}(x_i) + \rho \left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i) \right) && \text{for all } c \in C, i : i \in c, x_i \\ \beta_{ci}(x_c) &\leftarrow \beta_{ci}(x_c) + \rho \left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c) \right) && \text{for all } c \in C, i : i \in c, x_c \end{aligned}$$

References

[1] Afonso, M., Bioucas-Dias, J., Figueiredo, M.: Fast image recovery using variable splitting and constrained optimization. IEEE Transactions on Image Processing 19(9), 2345–2356 (2010)
[2] Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and distributed computation: numerical methods. Prentice-Hall, Inc., Upper Saddle River (2003)
[3] Duchi, J., Shalev-Shwartz, S., Singer, Y., Chandra, T.: Efficient projections onto the l1-ball for learning in high dimensions. In: Proceedings of the 25th International Conference on Machine learning, pp. 272–279 (2008)
[4] Eckstein, J., Bertsekas, D.P.: On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. Mathematical Programming 55, 293–318 (1992)
[5] Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. Computers and Mathematics with Applications 2, 17–40 (1976)
[6] Gimpel, K., Smith, N.A.: Softmax-margin crfs: training log-linear models with cost functions. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 733–736 (2010)
[7] Globerson, A., Jaakkola, T.: Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In: Advances in Neural Information Processing Systems, pp. 553–560 (2008)
[8] Glowinski, R., Marrocco, A.: Sur lapproximation, par elements finis dordre un, et la resolution, par penalisation-dualité, dune classe de problems de dirichlet non lineares. Revue Française d'Automatique, Informatique, et Recherche Opérationelle 9, 41–76 (1975)
[9] Goldfarb, D., Ma, S., Scheinberg, K.: Fast alternating linearization methods for minimizing the sum of two convex functions. Technical report, UCLA CAM (2010)
[10] Hazan, T., Shashua, A.: Norm-product belief propagation: Primal-dual message-passing for approximate inference. IEEE Transactions on Information Theory 56(12), 6294–6316 (2010)
[11] He, B.S., Yang, H., Wang, S.L.: Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. Journal of Optimization Theory and Applications 106, 337–356 (2000)
[12] Johnson, J.: Convex Relaxation Methods for Graphical Models: Lagrangian and Maximum Entropy Approaches. PhD thesis, EECS, MIT (2008)
[13] Jojic, V., Gould, S., Koller, D.: Fast and smooth: Accelerated dual decomposition for MAP inference. In: Proceedings of International Conference on Machine Learning (2010)
[14] Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(10), 1568–1583 (2006)
[15] Komodakis, N., Paragios, N.: Beyond loose LP-relaxations: Optimizing mRFs by repairing cycles. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 806–820. Springer, Heidelberg (2008)
[16] Komodakis, N., Paragios, N., Tziritas, G.: Mrf energy minimization and beyond via dual decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence 33, 531–552 (2011)

[17] Martins, A.F.T., Figueiredo, M.A.T., Aguiar, P.M.Q., Smith, N.A., Xing, E.P.: An augmented lagrangian approach to constrained map inference. In: International Conference on Machine Learning (June 2011)
[18] Meshi, O., Sontag, D., Jaakkola, T., Globerson, A.: Learning efficiently with approximate inference via dual losses. In: Proceedings of the 27th International Conference on Machine Learning, pp. 783–790 (2010)
[19] Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming 103, 127–152 (2005)
[20] Ravikumar, P., Agarwal, A., Wainwright, M.: Message-passing for graph-structured linear programs: proximal projections, convergence and rounding schemes. In: Proc. of the 25th International Conference on Machine Learning, pp. 800–807 (2008)
[21] Rush, A.M., Sontag, D., Collins, M., Jaakkola, T.: On dual decomposition and linear programming relaxations for natural language processing. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP (2010)
[22] Schlesinger, M.I.: Syntactic analysis of two-dimensional visual signals in noisy conditions. Kibernetika 4, 113–130 (1976)
[23] Sontag, D., Globerson, A., Jaakkola, T.: Introduction to dual decomposition for inference. In: Sra, S., Nowozin, S., Wright, S.J. (eds.) Optimization for Machine Learning. MIT Press, Cambridge (2011)
[24] Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T., Weiss, Y.: Tightening LP relaxations for MAP using message passing. In: Proc. of the 24th Annual Conference on Uncertainty in Artificial Intelligence, pp. 503–510 (2008)
[25] Taskar, B., Guestrin, C., Koller, D.: Max margin Markov networks. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems, vol. 16, pp. 25–32. MIT Press, Cambridge (2004)
[26] Tosserams, S., Etman, L., Papalambros, P., Rooda, J.: An augmented lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers. Structural and Multidisciplinary Optimization 31, 176–189 (2006)
[27] Wainwright, M.J., Jordan, M.: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning 1(1-2), 1–305 (2008)
[28] Werner, T.: A linear programming approach to max-sum problem: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 1165–1179 (2007)
[29] Yanover, C., Meltzer, T., Weiss, Y.: Linear programming relaxations and belief propagation – an empirical study. Journal of Machine Learning Research 7, 1887–1907 (2006)

Aggregating Independent and Dependent Models to Learn Multi-label Classifiers*

Elena Montañés, José Ramón Quevedo, and Juan José del Coz

Artificial Intelligence Center, University of Oviedo at Gijón (Spain)
{elena,quevedo,juanjo}@aic.uniovi.es

Abstract. The aim of multi-label classification is to automatically obtain models able to tag objects with the labels that better describe them. Despite it could seem like any other classification task, it is widely known that exploiting the presence of certain correlations between labels helps to improve the classification performance. In other words, object descriptions are usually not enough to induce good models, also label information must be taken into account. This paper presents an aggregated approach that combines two groups of classifiers, one assuming independence between labels, and the other considering fully conditional dependence among them. The framework proposed here can be applied not only for multi-label classification, but also in multi-label ranking tasks. Experiments carried out over several datasets endorse the superiority of our approach with regard to other methods in terms of some evaluation measures, keeping competitiveness in terms of others.

1 Introduction

In multi-label classification the goal is to induce a hypothesis to assign a set of labels for each instance rather than a single class, as happens in multi-class classification. This kind of tasks arises in many practical domains; nowadays almost all media contents (text documents, songs, movies or videos) are tagged with several labels to briefly inform users about their actual content. Another well-known example in the research community is the keywords attached to a paper; useful to indicate the *relevant* topics of the paper.

At first sight, one could think that multi-label classification can be easily solved applying or adapting state-of-the-art (binary or multi-class) classification algorithms. In fact, many of the first approaches proposed were aimed to extend these methods to handle multi-label data, including decision trees [2], instance-based algorithms [22], Neural Networks [21], Support Vector Machines [6], Naive Bayes [12], Conditional Random Fields [9] and boosting [16]. However, in order to obtain good performance results, it is not enough to adapt a good learning approach, otherwise it is also necessary to design specific methods that exploit somehow the particularities of multi-label data, as most of the cited works do.

* This research has been partially supported by Spanish Ministerio de Ciencia e Innovación (MICINN) grant TIN2008-06247.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 484–500, 2011.

Mainly, multi-label learning presents two challenging problems. The first one bears on the computational complexity of the algorithms. If the number of labels is high, then a very complex approach is not practical for business use, so the scalability is a key issue in this field. The second dare is related with the own nature of this task and the multi-label data. Not only the number of classes is higher than in multi-class classification, but also each example belongs to an indeterminate number of labels, and more important, labels present some relationships between them. In other words, object descriptions are usually not enough to induce correct models, also the information among labels must be taken into account. The dimensionality of the label space, together with the possibility of incomplete labeling data obtained from different sources, make this goal even more difficult to achieve.

Despite the first issue is important to make algorithms applicable in large domains, from a learning perspective probably the hottest topic in multi-label community is to design new methods able to detect and exploit dependences among labels. In fact, several methods are being proposed in that direction. Roughly speaking, we can categorize them according to two major properties: i) the size of the subset of labels whose dependences are searched for; and ii) depending on the type of correlations they try to find. Looking at the first property, we have those methods that only consider pairwise relations between labels [6,7,13,16,21], and, secondly, approaches that take into account correlations among the labels in bigger subsets [14,15,19], including those that consider the influence of all other labels in the prediction of one particular label [1,10]. On the other hand, concerning the type on dependences they seek to capture [3], there are some methods designed to detect conditional label dependence (refered to the dependence of the labels given a specific instance), for example [3,9,15,18]; and unconditional dependence (a global type of dependence independent of any concrete observation), see [1,10,21].

The main proposal of this paper is grounded on the following idea. Thinking again about the keywords example, all of us know that, for a particular paper, there are some clearly relevant and irrelevant keywords. But there are also a few keywords than can be considered relevant or irrelevant, depending on the authors' opinion. Different authors with different criteria could select some different keywords for the same paper. Our assumption is that labels that are clearly relevant or clearly irrelevant can be predicted using only the description of the object. However, considering the relationships with other labels is also necessary to correctly assign the somehow-relevant labels. This is the reason why this paper presents a decomposition approach based on aggregating two groups of classifiers: the first one is learned assuming label independence, and the second one is built considering a complete label dependence. These dependent classifiers are designed to capture conditional dependence, taking into account the influence of all other labels. Another interesting property of the approach presented here is that it can be easily adapted to cope with different multi-label learning tasks, not only classification, but also ranking.

The rest of the paper is organized as follows. Next section describes a formal framework for multi-label classification and reviews previous approaches related with this work. Section 3 exposes the proposals of this paper, based on the idea of combining two groups of binary models. Finally, experimental results are reported in Section 4 and some conclusions are drawn in Section 5.

2 Notation and Related Work

2.1 Formal Framework for Multi-label Classification

Let $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_m\}$ be a finite and non-empty set of labels, and let $\mathcal{X}$ be an input space. We consider a multi-label classification task given by a training set $S = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$, whose instances were independently and randomly obtained from an unknown probability distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ on $\mathcal{X} \times \mathcal{Y}$, in which the output space $\mathcal{Y}$ is the power set of $\mathcal{L}$, in symbols $\mathcal{P}(\mathcal{L})$. In order to make the notation easier to read, we define $\boldsymbol{y}_i$ as a binary vector, $\boldsymbol{y}_i = \{y_1, y_2, \ldots, y_m\}$, in which each component $y_j = 1$ indicates the presence of label ℓ_j in the set of relevant labels of $\boldsymbol{x}_i$. Using this convention, the output space can be also defined as $\mathcal{Y} = \{0, 1\}^m$.

The goal of a multi-label classification is to induce a hypothesis $\boldsymbol{h} : \mathcal{X} \longrightarrow \mathcal{Y}$ from S, that correctly predicts the subset of labels from $\mathcal{L}$ for a new unlabeled instance $\boldsymbol{x}$. Without any loss of generality, this hypothesis can be seen as a combination of a collection of sub-hypotheses, $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots h_m(\boldsymbol{x}))$, one per label, in which each h_j takes the form of

$$h_j : \mathcal{X} \longrightarrow \{0, 1\}, \tag{1}$$

and it is able to predict if the label ℓ_j must be attached to the instance $\boldsymbol{x}$ or not. Sometimes, the goal is not to return the relevant labels, but to obtain the posterior probability of each label given $\boldsymbol{x}$. In that case, $h_j : \mathcal{X} \longrightarrow [0, 1]$. Notice that these probabilistic classifiers can also be useful to rank the labels according to their posteriors. This paper only discusses some multi-label classifiers of the form of Equation 1, despite many of them can have a probabilistic/ranking version, including ours.

In order to measure the performance of multi-label classifiers, several metrics have been proposed. A unified presentation of existing evaluation measures for multi-label classification can be found in [18], including their categorization into example-based, label-based and ranking-based measures. Here we will consider only example-based evaluation metrics for three reasons: i) this paper focuses on multi-label classification rather than multi-label ranking, moreover, some of the state-of-the-art methods employed to compare do not produce a ranking of labels, ii) we are interested in studying whether the different approaches capture or not the dependencies among labels at example-level, and iii) some of the example-based metrics used were originally proposed in [10], that also introduces a stacking-based method for multi-label learning, which is probably the most similar approach to ours. The following evaluation measures have been taken from the Information Retrieval field:

– **Jaccard index** computes the percentage of relevant labels predicted in the subset formed by the union of returned and relevant labels[1],

$$Jaccard(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![y_i = 1 \text{ or } h_i(\boldsymbol{x}) = 1]\!]}. \tag{2}$$

– **Precision** determines the fraction of relevant labels in the predicted labels,

$$Precision(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![h_i(\boldsymbol{x}) = 1]\!]}. \tag{3}$$

– **Recall** is the proportion of relevant labels of the example correctly predicted,

$$Recall(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![y_i = 1]\!]}. \tag{4}$$

– $\boldsymbol{F_1}$ is the evenly weighted harmonic mean of Precision and Recall,

$$F_1(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{2 \sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} ([\![y_i = 1]\!] + [\![h_i(\boldsymbol{x}) = 1]\!])}. \tag{5}$$

The evaluation metrics presented above are biased towards those methods that correctly predict the relevant labels. That is one of the reasons to select them, because they allow us to study if relevant labels are detected or not, specially in those situations where some kind of correlation occurs. Finally, the performance in multi-label classification can be reported in terms of other two measures:

– **Hamming loss**, which is defined as the proportion of labels whose relevance is incorrectly predicted:

$$Hamming(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq h_i(\boldsymbol{x})]\!]. \tag{6}$$

– **0/1 loss**, looks if predicted and relevant label subsets are equal or not.

$$Zero - One(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = [\![\boldsymbol{y} \neq \boldsymbol{h}(\boldsymbol{x})]\!]. \tag{7}$$

2.2 Some Approaches for Multi-label Classification

The most employed baseline method for multi-label classification is the Binary Relevance (BR) algorithm. BR decomposes the learning of $\boldsymbol{h}$ into a set of binary classification tasks, one per label, where each single model h_j is learned independently of the rest, using only the information of that particular label and ignoring all other labels. Despite its simplicity, BR algorithm presents several advantages: i) any binary learning method can be taken as base learner, ii) it has linear complexity with respect to the number of labels and iii) it can be easily parallelized. The main drawback of BR is that it does not take into account

[1] The expression $[\![p]\!]$ evaluates to 1 if the predicate p is true, and to 0 otherwise.

any label dependences and may fail to predict some label combinations if such correlations are present. However, using a state-of-the-art base learner, for instance SVM, with a proper tuning parameters process, BR usually obtains quite good results in benchmark datasets of the literature. Moreover, for some particular evaluation metrics, as Hamming loss (Eq. 6), BR offers a very competitive performance. This behavior can be explained studying BR from a probabilistic point of view. As in most classification learning process, each binary model h_j is able to estimate $\mathbf{P}(y_j|\boldsymbol{x})$. This is the reason why BR is well-suited for every loss function whose risk minimizer can be expressed in terms of marginal distributions of labels, for instance Hamming loss. On the other hand, the fact that BR does not take label dependence into account obviously affects its performance for evaluation measures like 0/1 loss. Hence, it is necessary to estimate the joint label probability distributions to obtain predictions that minimize this sort of metrics. A formal probabilistic analysis of multi-label classification, studying the connection between risk minimization and loss functions can be found in [3,4].

Godbole and Sharawagi present one approach [10] to overcome the label-independence problem of BR. They apply the stacked generalization learning paradigm [20], also known simply as stacking, in the context of multi-label classification. In the learning phase, their method builds a stack of two groups of classifiers. The first one is formed by the same binary classifiers yielded by BR method, in symbols, $\boldsymbol{h}^1(\boldsymbol{x}) = (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x}))$. In a second level, also called meta-level, another group of binary models (one for each label again) is learned, but these classifiers consider an augmented feature space that includes the binary outputs of all models of the first level, $\boldsymbol{h}^2(\boldsymbol{x}, \boldsymbol{y}') = (h_1^2(\boldsymbol{x}, \boldsymbol{y}'), \ldots, h_m^2(\boldsymbol{x}, \boldsymbol{y}'))$, where $\boldsymbol{y}' = \boldsymbol{h}^1(\boldsymbol{x})$. The idea is to learn the relationships between labels in the meta-level step. In the testing phase, the final predictions are the outputs of the meta-level classifiers, $\boldsymbol{h}^2(\boldsymbol{x})$, using the outputs of $\boldsymbol{h}^1(\boldsymbol{x})$ exclusively to obtain the values of the augmented feature space.

Some variants of the stacking approach have been proposed, mainly focused on reducing the augmented feature space removing some label dimensions. The idea is to ignore the information of those labels that are no related with the label j of the model h_j^2 being learned. For instance, in [17] the authors propose to calculate the chi-coefficient between each pair of labels, (j, k), based on an initial single pass over the training set. The method prunes the information of each label k with a correlation below a threshold to induce the meta-model h_j^2. This approach improves the computational efficiency, without any significant loss in predictive performance, even some gains are obtained for some data sets.

In any case, the meta-level classifiers of the stacking approach estimates $\mathbf{P}(y_j|\boldsymbol{x}, \boldsymbol{y}')$, where $\boldsymbol{y}'$ is in turn an estimation that depends only on $\boldsymbol{x}$. This chain of estimations can explain why perhaps $\boldsymbol{y}'$ does not contain enough information to infer the dependence of label y_j with respect to other labels.

Read et al. describe [15] a learning algorithm called Classifier Chain (CC), that can model label correlations while maintaining a computational complexity of the same order as that of BR. As its name denotes, CC involves m binary classifiers linked along a chain, where each classifier deals with the binary relevance problem

associated with one label. In the training phase, the feature space of each classifier in the chain is extended with the actual label information of all previous links. For instance, if the chain follows the order of the set of labels, then the functional form of each classifier h_j will be:

$$h_j : \mathcal{X} \times \{0,1\}^{j-1} \longrightarrow \{0,1\}, \tag{8}$$

in which the actual label data of the previous labels in the chain, $y_1, \ldots, y_{j-1}$, are used to build h_j. Notice that all binary models can be learned in parallel. However, in the testing phase, the classifiers are applied following the chain order, using the binary outputs of the previous models as additional input information. In symbols, $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, h_1(\boldsymbol{x})), h_3(\boldsymbol{x}, h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, h_1(\boldsymbol{x}))), \ldots)$. Obviously, the label order in the chain affects the performance obtained. Although heuristics can be used to select a promising chain order, the authors solve the issue by an Ensemble of Classifier Chains (ECC). This approach ensembles different random chain orderings and a different sample of the training data for learning each CC model.

Dembczyński et al. present in [3] a probabilistic framework for multi-label classification and propose the Probabilistic Classifier Chains (PCC) and its ensemble version (EPCC). They generalize and outperform their counterpart methods, CC and ECC, but increasing their testing complexity. One of the interesting contributions of the paper is that it offers a probabilistic interpretation of CC. Given an instance $\boldsymbol{x}$ it is possible to compute the conditional probability of each label combination $\boldsymbol{y} \in \mathcal{Y}$, applying the product rule of probability:

$$\mathbf{P}(\boldsymbol{y}|\boldsymbol{x}) = \mathbf{P}(y_1|\boldsymbol{x}) \prod_{j=2}^{m} \mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}), \tag{9}$$

whose probabilities can be obtained from the classifiers of the chain (Eq. 8) when a probabilistic learner is used. The difference between PCC and CC is that the former estimates the entire joint distribution of labels, whereas the later takes sequential decisions and, in that sense, it offers a deterministic approximation of PCC. PCC produces much better estimates but at the cost of higher complexity, limiting its applicability to data sets with a small number of labels.

There are other methods, less related with our approach, that try to find interdependences between labels. RAkEL(RAndom k-labELsets), presented by Tsoumakas and Vlahavas [19], iteratively constructs an ensemble of Label Powerset (LP) classifiers. LP algorithm considers each unique subset of labels that exists in a multi-label training set as one of the classes of a new multi-class classification task. At each iteration, RAkEL randomly selects a k-labelset $\boldsymbol{Y}_i$ from $\mathcal{L}$ without replacement. Then, it learns a LP classifier of the form $\mathcal{X} \rightarrow \mathcal{P}(\boldsymbol{Y}_i)$. A simple voting process determines the final classification set. Also, Cheng and Hüllermeier propose IBLR (Instance-Based Learning by Logistic Regression) [1]. IBLR unifies instance-based learning and logistic regression, comprising both methods as special cases. Considering only the label dependence problem, the main idea is to extend the description of each example $\boldsymbol{x}$ by additional features that express the presence or absent of each label in the neighborhood of $\boldsymbol{x}$.

3 Aggregating Independent and Dependent Classifiers

The main proposal of this paper is to build a multi-label classifier that combines the two main options to tackle multi-label learning. On one hand, there are methods based on the assumption of label independence, that is, they only use object descriptions in order to predict the labels attached to the object, as BR. On the other hand, there are plenty of algorithms that induce models considering some kind of label dependence, that is, they also employ the information about other labels, like those described in the previous section. Formally, the latter approach can encapsulate the former, in the sense that a label-dependent model can also capture the cases that an independent model predicts well. However, when the interdependences among labels are complex, learning reliable dependent models becomes more difficult: richest hypothesis spaces must be used, increasing the risk of overfitting, and more labeled data is needed, which will not be available in some cases.

In our opinion, the two approaches are not exclusive, but complementary. Hence, aggregating them may produce a more robust multi-label classifier. This is the reason to propose Aggregating Independent and Dependent classifiers (AID), a decomposition method that combines two groups of models. The first one is learned assuming label independence and it will be formed by the same classifiers yielded by BR method or by the first-level models of the stacking-based approach [10], $\boldsymbol{h}^1(\boldsymbol{x}) = (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x}))$. The second group of binary classifiers, $\boldsymbol{h}^2(\boldsymbol{x}, \boldsymbol{y}) = (h_1^2(\boldsymbol{x}, y_2, \ldots, y_m), \ldots, h_m^2(\boldsymbol{x}, y_1, \ldots, y_{m-1}))$, is built considering the information of all other labels. Thus, each h_j^2 is defined as:

$$h_j^2 : \mathcal{X} \times \{0,1\}^{m-1} \longrightarrow \{0,1\}. \tag{10}$$

These classifiers try to detect fully conditional label dependence. Notice that all models can be learned in parallel, because they are induced using only training data. However, in the testing phase, the classifiers of the first group, $\boldsymbol{h}^1$, are applied first and their binary outputs form the label features of models $\boldsymbol{h}^2$. The final prediction is calculated aggregating both groups of responses:

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{x}) = \oplus(& (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x})), \\ & (h_1^2(\boldsymbol{x}, h_2^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x})), \ldots, h_m^2(\boldsymbol{x}, h_1^1(\boldsymbol{x}), \ldots, h_{m-1}^1(\boldsymbol{x}))) \ ), \end{aligned} \tag{11}$$

in which $\oplus$ can be selected by practitioners, depending on the target loss function and on the specific learning task. A natural choice for multi-label classification is the $or()$ function, being $max()$ for ranking whenever $\boldsymbol{h}^1$ and $\boldsymbol{h}^2$ provide probabilities. From a probabilistic point of view, AID method merges two different estimations $\mathbf{P}(y_j|\boldsymbol{x})$ and $\mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_m)$ for label y_j. In this sense AID takes into account the conditional dependence between labels.

Let us highlight that our method is quite general and can be adapted and improved in several directions. We will cite three of them: i) the aggregate function $\oplus$ can be selected (or even learned) to optimize a specific target loss function, ii) for a given query, the dependent models $\boldsymbol{h}^2$ can be iteratively applied until not

new labels are assigned, as it is pointed out in [10], and iii) some dimensions of the label data can be removed in order to make the learning of models $\boldsymbol{h}^2$ easier, for instance, applying methods like [17]. Due to the lack of space, the study of all these issues are beyond the scope of this work.

3.1 Comparison with Related Approaches

In order to better understand the properties of our method, it is interesting to analyze the differences with respect to the most related approaches [3,10,15] described above. We support the idea that AID solves two drawbacks of the stacking-based approach [10]. Firstly, the independent models learned in the first level are only employed to obtain the label-related features for training and testing the meta-level classifiers. In other words, they are not used as predictors, when it is well-known that BR classifiers by themselves can obtain a relatively good performance. The outputs of independent classifiers, once they are calculated, can be additionally employed to decide the predicted labels. Secondly, and even more important, maybe some information about the dependence among labels is missing when learning the meta-level classifiers. Instead of using the actual labels of each example to augment the feature space, as AID classifier does, the stacking method employs first-level classifiers predictions. Although it is absolutely formal from a learning perspective–the data source is the same in both training and the testing phases for meta-level classifiers–, if we think about the trueness of the training data, the actual labels are less noisy and contain the true correlation information among the labels. For this reason the estimation of $\mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_m)$ is expected to be more accurate than $\mathbf{P}(y_j|\boldsymbol{x}, \boldsymbol{y}')$. Next section experimentally analyzes these two issues.

Comparing our method with Classifier Chain [15], including its probabilistic variant (PCC) [3], we can found pros and cons for both. One of the best properties of CC approaches, specially PCC, is that they are supported by probability theory. PCC is able to estimate the entire joint distribution of the labels and can select the most appropriate label combination for a particular loss function, whereas CC and our approach offer only greedy approximations. But, on the other hand, CC variants assume an in-chain dependence between labels when sometimes their interdependences are much more complex. Moreover, some probability estimations of the chain can be poor when the correlated labels of a label are not placed before in the chain, which most likely happens at the first links of the chain. This is the reason for ensembling several CC or PCC classifiers. Our approach considers for every label all others, so the correlated labels are always taken into account, although detecting their dependences can be more difficult.

Analyzing the training computational complexity, all the approaches incur in the same linear complexity with respect to the number of labels, but they differ in the number of models, CC/PCC (m), stacking method and AID ($2m$), and ECC/EPCC (Nm, where N is the number of ensembles). The testing complexity of AID, the stacking method and CC/ECC is linear again, differing in the number of evaluations requiered. PCC/EPCC evaluate an exponential order times the models limiting their applicability to domains with few labels.

Table 1. Properties of the data sets used in the experiments

Data set	Attributes	Examples	Labels	Cardinality
bibtex	1836	7395	159	2.40
emotions	72	593	6	1.87
enron	1001	1702	53	3.38
genbase	1185	662	27	1.25
image	135	2000	5	1.24
mediamill	120	5000	101	4.27
medical	1449	978	45	1.25
reuters	243	7119	7	1.24
scene	294	2407	6	1.07
slashdot	1079	3782	22	1.18
yeast	103	2417	14	4.24

4 Experiments

This section reports the results of the experiments performed to evaluate the proposed multi-label classification method. The aim of the experiments was twofold. Firstly, a deep comparative study between aggregating-based and stacking-based methods was performed. The idea was to experimentally analyze the different properties of both kind of strategies. Secondly, our aggregating approach was compared with some other state-of-the-art methods for multi-label classification, most of them aimed to detect correlations among labels. The experiments were performed over several multi-label data sets whose main properties are shown in Table 1. As it can be seen, they are quite different among them in the number of attributes, examples, labels and cardinality (number of labels per example).

We tested two groups of multi-label classification algorithms. In the first place, AID classifier (using $or()$ as the aggregate function) and stacking-based approach, denoted as STA, were compared. We also included a couple of variants of both to study some of their properties. Then, our aggregating approach was compared with BR, MLkNN [22], RAkEL [19], IBLR [1] and ECC [15], in the version described in [3], named as ECC*. Among chain-based methods, we selected ECC* because it offers the best trade-off between performance and complexity [3]. CC performs worse and PCC/EPCC have a higher computational complexity.

The binary base learner employed to obtain single classifiers for each label was the *logistic regression* of [11]. The regularization parameter C was established for each binary model performing a grid search over the values $C \in \{10^p \mid p \in [-3, \ldots, 3]\}$ optimizing the accuracy estimated by means of a balanced 2-fold cross validation repeated 5 times. Such parameter settings have only sense for BR, ECC* and AID, and kept equal for all of these methods. That is, their models are exactly the same when their respective input spaces are equal. The parameters taken for the rest of the state-of-the-art methods were the default ones suggested by their authors.

The evaluation measures applied were those discussed in Section 2.1. As they are defined on a per instance basis, the value for a test set is the average over all instances. The scores reported, displayed as percentages for all measures, were estimated by means of a 10-fold cross-validation. The ranks of each data sets are indicated in brackets. In case of ties, average ranks are shown. The average ranks over all data sets are computed and shown at the last row of each table. Following the recommendations of [5] and [8], a two-step comparison for each of the considered measures was performed. The first step is a Friedman test that rejects the null hypothesis that states that not all learners perform equally. The second step is a *post-hoc* pairwise comparison. We performed a Bergmann-Hommel's test using the software provided in [8]. This comparison is preferred to that of Nemenyi [5], because it is a less conservative procedure able to detect certain obvious differences that Nemenyi's test may not obtain. In any case, the significant differences found with both tests are almost equal.

4.1 AID Classifier vs. Stacking Approach

The goal of this experiment was to compare AID classifier and STA method and also to gain some insights about their principles. This was the motivation to include two variants of both. Table 2 and Table 3 show the scores of all them; BR was added as the baseline reference. Our main intention was to obtain answers to the following questions: i) Which method performs better, AID or STA? ii) Which information is more appropriate to detect the correlations among labels, the actual label data or the predictions of independent models? iii) In the testing phase, which data is preferable to augment the feature space of dependent models, binary or probabilistic outputs? And, finally, iv) is preferable to aggregate or to stack the predictions of independent and dependent models?

The first question is the easiest to answer because it only involves two of the algorithms. AID classifier performs better than STA in all the measures except in Hamming loss. The differences are significant in the cases of Recall, F_1 and Jaccard index (see Table 6). In fact, our proposed method ranks first in five out the six performance measures, while STA is not in the top-three for any of them. Comparing both with the baseline method (BR), AID obtains better scores in all metrics (significant differences in Recall, F_1 and Jaccard index) except in Hamming loss, in which BR performs significantly better. In the comparison between STA and BR, STA wins in four out of the six (except Hamming loss and Precision), but the differences are quite small and none of them are significant.

In order to answer the second question we included two new methods. The idea was to feed AID classifier and STA with the training information that uses the other one to learn the dependent models, $\boldsymbol{h}^2$. Following this idea, AID$^{\boldsymbol{y}'}$ classifier uses the predicted labels given by independent models $\boldsymbol{h}^1$, and STA$^{\boldsymbol{y}}$ employs the true label data. Comparing now each algorithm with its counterpart (AID vs. AID$^{\boldsymbol{y}'}$, STA vs. STA$^{\boldsymbol{y}}$), we find that in both cases the algorithm that uses the actual label information (AID or STA$^{\boldsymbol{y}}$) improves the performance of its counterpart in all measures, but Hamming loss. The differences are significant in the case of F_1 and Jaccard index for AID vs. AID$^{\boldsymbol{y}'}$. Interestingly, now AID$^{\boldsymbol{y}'}$

Table 2. Aggregated vs. stacking-based approaches: Precision, Recall, F_1 and Jaccard index

Precision	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	48.19(3)	48.70(2)	47.17(7)	48.01(4.5)	48.01(4.5)	**49.08**(1)	47.39 (6)
Emotions	56.36(6)	**62.30**(1)	62.09(2)	57.99(4)	56.58(5)	59.03(3)	54.31 (7)
Enron	**69.99**(1)	66.10(3)	65.51(4)	65.05(6)	65.18(5)	66.37(2)	61.69 (7)
Genbase	99.52(4)	99.52(4)	**99.60**(1.5)	99.40(6.5)	99.40(6.5)	99.52(4)	**99.60** (1.5)
Image	44.23(7)	**53.97**(1)	53.88(2)	44.49(6)	44.54(5)	48.18(3)	46.17 (4)
Mediamill	**78.81**(1)	70.91(3)	70.11(5)	44.06(6)	43.49(7)	70.16(4)	71.67 (2)
Medical	78.94(6)	**82.50**(1)	81.33(2)	79.47(4.5)	79.47(4.5)	80.88(3)	78.31 (7)
Reuters	85.79(7)	**87.53**(1)	87.50(2)	85.88(6)	85.89(5)	87.48(3)	86.39 (4)
Scene	61.46(7)	67.81(3)	66.14(6)	66.88(5)	67.13(4)	**71.89**(1)	69.05 (2)
Slashdot	46.06(6)	**53.32**(1)	53.20(2)	47.91(3.5)	47.91(3.5)	46.98(5)	23.92 (7)
Yeast	**71.13**(1)	66.50(7)	66.80(6)	70.82(2)	70.81(3)	67.83(5)	68.51 (4)
Avg. rank	(4.45)	(**2.45**)	(3.59)	(4.91)	(4.82)	(3.09)	(4.68)
Recall	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	33.80(7)	**36.86**(1)	34.82(5)	35.50(3)	35.49(4)	36.47(2)	34.06 (6)
Emotions	48.16(6)	**68.03**(1)	65.84(2)	51.02(4)	49.81(5)	53.44(3)	47.00 (7)
Enron	50.50(6)	**59.89**(1)	57.48(2)	56.78(3)	56.68(4)	54.86(5)	41.78 (7)
Genbase	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07** (4)
Image	43.32(6)	**55.96**(1)	53.68(2)	43.69(4)	43.54(5)	47.74(3)	43.07 (7)
Mediamill	52.33(7)	59.25(3)	54.34(5)	**62.03**(1)	60.02(2)	58.02(4)	52.41 (6)
Medical	78.34(6)	**83.86**(1)	81.02(4)	83.09(2.5)	83.09(2.5)	80.57(5)	75.09 (7)
Reuters	84.90(7)	**91.90**(1)	90.29(2)	85.01(5)	84.93(6)	87.90(3)	85.10 (4)
Scene	62.87(7)	**87.55**(1)	82.38(2)	68.50(5)	68.17(6)	77.10(3)	70.81 (4)
Slashdot	44.21(6)	**71.42**(1)	70.37(2)	45.96(3.5)	45.96(3.5)	45.05(5)	21.68 (7)
Yeast	58.86(6)	**66.42**(1)	60.93(3)	59.43(4)	59.38(5)	62.50(2)	55.83 (7)
Avg. rank	(6.18)	(**1.45**)	(3.00)	(3.55)	(4.27)	(3.55)	(6.00)
$\boldsymbol{F_1}$	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	37.02(7)	**39.22**(1)	37.54(5)	37.93(3)	37.92(4)	39.03(2)	37.04 (6)
Emotions	49.20(6)	**62.05**(1)	60.87(2)	51.54(4)	50.33(5)	53.31(3)	48.03 (7)
Enron	55.66(6)	**59.97**(1)	58.20(2)	57.78(3.5)	57.78(3.5)	56.73(5)	47.32 (7)
Genbase	99.18(4)	99.18(4)	**99.21**(1.5)	99.10(6.5)	99.10(6.5)	99.18(4)	**99.21** (1.5)
Image	42.12(7)	**52.69**(1)	51.68(2)	42.42(5)	42.38(6)	46.18(3)	43.10 (4)
Mediamill	59.17(3)	**60.39**(1)	57.23(4)	48.00(6)	47.06(7)	59.67(2)	56.67 (5)
Medical	77.33(6)	**81.66**(1)	79.82(2)	79.45(3.5)	79.45(3.5)	79.35(5)	75.54 (7)
Reuters	84.13(7)	**87.74**(1)	87.05(2)	84.23(5)	84.21(6)	86.43(3)	84.65 (4)
Scene	61.25(7)	71.38(2)	68.46(4)	66.68(6)	66.75(5)	**72.85**(1)	68.78 (3)
Slashdot	44.33(6)	**55.97**(1)	55.47(2)	46.05(3.5)	46.05(3.5)	45.18(5)	22.36 (7)
Yeast	61.68(5)	**63.28**(1)	60.98(6)	61.89(3)	61.86(4)	62.42(2)	58.44 (7)
Avg. rank	(5.82)	(**1.36**)	(2.95)	(4.45)	(4.91)	(3.18)	(5.32)
Jaccard	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	31.50(7)	**33.59**(1)	32.32(3)	32.16(4)	32.15(5)	33.28(2)	31.70 (6)
Emotions	42.27(6)	**52.89**(1)	51.76(2)	44.52(4)	43.46(5)	46.12(3)	41.68 (7)
Enron	44.69(5)	**48.50**(1)	47.09(2)	45.95(3.5)	45.95(3.5)	44.23(6)	35.81 (7)
Genbase	98.94(4)	98.94(4)	**98.97**(1.5)	98.82(6.5)	98.82(6.5)	98.94(4)	**98.97** (1.5)
Image	38.60(7)	**47.88**(1)	47.32(2)	38.85(6)	38.87(5)	42.43(3)	39.95 (4)
Mediamill	46.70(2)	**48.13**(1)	45.42(4)	34.70(6)	34.09(7)	46.00(3)	42.47 (5)
Medical	74.51(6)	**78.40**(1)	76.95(2)	75.43(4.5)	75.43(4.5)	76.37(3)	73.13 (7)
Reuters	81.67(7)	**84.37**(1)	84.00(2)	81.77(5)	81.76(6)	83.80(3)	82.36 (4)
Scene	59.41(7)	65.90(3)	64.00(6)	64.64(5)	64.88(4)	**69.74**(1)	66.67 (2)
Slashdot	42.71(6)	**49.98**(1)	49.70(2)	44.29(3.5)	44.29(3.5)	43.50(5)	21.55 (7)
Yeast	50.71(5)	**52.40**(1)	49.68(6)	50.97(3)	50.93(4)	51.22(2)	46.77 (7)
Avg. rank	(5.64)	(**1.45**)	(2.95)	(4.64)	(4.91)	(3.18)	(5.23)

Table 3. Aggregated vs. stacking-based approaches: Hamming loss and 0/1 loss

Hamming	BR	AID	STA$^{\boldsymbol{y}}$	AID$^{\boldsymbol{y}'}$	STA	AID$^{\boldsymbol{p}}$	STA$^{\boldsymbol{p}}$
Bibtex	**1.21**(1.5)	1.22(4)	**1.21**(1.5)	1.26(6.5)	1.26(6.5)	1.22(4)	1.22 (4)
Emotions	22.03(4)	23.04(6)	23.15(7)	21.75(2)	21.83(3)	**21.47**(1)	22.14 (5)
Enron	**4.46**(1)	4.82(3)	4.88(6)	4.84(5)	4.83(4)	4.69(2)	4.95 (7)
Genbase	0.08(4)	0.08(4)	**0.07**(1.5)	0.09(6.5)	0.09(6.5)	0.08(4)	**0.07** (1.5)
Image	20.25(3.5)	21.90(7)	21.61(6)	20.33(5)	20.23(2)	20.25(3.5)	**20.04** (1)
Mediamill	**2.76**(1)	3.03(3)	3.10(5)	5.48(6)	5.50(7)	2.99(2)	3.09 (4)
Medical	0.99(4)	**0.95**(1)	0.98(3)	1.12(6.5)	1.12(6.5)	0.97(2)	1.02 (5)
Reuters	4.58(3)	5.81(7)	5.77(6)	4.59(4)	4.60(5)	**4.44**(1)	4.51 (2)
Scene	**9.83**(1)	18.66(7)	18.31(6)	9.99(3)	9.85(2)	10.28(5)	10.07 (4)
Slashdot	**3.73**(1)	8.81(7)	8.80(6)	3.86(3.5)	3.86(3.5)	3.75(2)	4.49 (5)
Yeast	**19.81**(1)	21.29(5)	21.56(7)	19.85(2.5)	19.85(2.5)	20.66(4)	21.41 (6)
Avg. rank	(**2.27**)	(4.91)	(5.00)	(4.59)	(4.41)	(2.77)	(4.05)
0/1 loss	BR	AID	STA$^{\boldsymbol{y}}$	AID$^{\boldsymbol{y}'}$	STA	AID$^{\boldsymbol{p}}$	STA$^{\boldsymbol{p}}$
Bibtex	82.83(5)	**81.49**(1)	81.54(2)	82.99(6.5)	82.99(6.5)	81.96(3)	82.30 (4)
Emotions	79.42(7)	**74.19**(1)	74.70(2)	77.40(4)	77.91(6)	76.05(3)	77.73 (5)
Enron	86.90(3)	85.37(2)	**85.25**(1)	88.13(5)	88.07(4)	92.54(6)	95.95 (7)
Genbase	**1.81**(3)	**1.81**(3)	**1.81**(3)	2.11(6.5)	2.11(6.5)	**1.81**(3)	**1.81** (3)
Image	71.50(7)	65.75(2)	**65.20**(1)	71.40(6)	71.25(5)	68.35(3)	69.20 (4)
Mediamill	90.36(3)	**88.10**(1)	88.14(2)	97.96(7)	97.90(6)	94.48(4)	96.54 (5)
Medical	33.94(4)	**31.18**(1)	31.49(2)	36.61(6.5)	36.61(6.5)	32.62(3)	34.05 (5)
Reuters	25.69(7)	25.16(4)	24.60(3)	25.59(6)	25.54(5)	**24.10**(1)	24.50 (2)
Scene	46.03(6)	46.07(7)	44.99(5)	41.38(4)	40.63(3)	**39.26**(1)	39.38 (2)
Slashdot	61.95(4)	63.20(6)	62.85(5)	**60.82**(1.5)	**60.82**(1.5)	61.34(3)	80.83 (7)
Yeast	84.53(5)	**83.08**(1)	84.07(4)	83.95(2.5)	83.95(2.5)	85.85(6)	90.03 (7)
Avg. rank	(4.91)	(**2.64**)	(2.73)	(5.05)	(4.77)	(3.27)	(4.64)

is only better than BR in Recall, but STA$^{\boldsymbol{y}}$ is significantly better than BR in Recall, F_1 and Jaccard index, and worse in Hamming loss. The results concerning Hamming loss are quite intriguing, because both approaches seem able to improve when true labels are used, but the results in terms of Hamming loss are worse. Despite this fact, we do think that these results confirm the idea that using the actual label data is better to capture the correlations among labels.

Given that the base learner employed is a logistic regressor that provides posterior probabilities, we included two other methods (AID$^{\boldsymbol{p}}$, STA$^{\boldsymbol{p}}$) that consist of taking the probabilities yielded by the independent models rather than their binary outputs in the testing phase, remaining unchanged the original training phase for each approach. Comparing their results with the original versions, we have that AID performs better than AID$^{\boldsymbol{p}}$ in all measures, except in Hamming loss in which the performance of AID$^{\boldsymbol{p}}$ ranks second behind BR. AID$^{\boldsymbol{p}}$ is significantly better than BR in Recall, F_1 and Jaccard index, and it is very competitive in Hamming loss. In the case of STA$^{\boldsymbol{p}}$ we do not observe any improvement, otherwise the results are pretty the same of STA. It seems that using the posteriors helps to reduce the Hamming loss, but it is worse for the other measures.

Finally, we want to compare the aggregating and stacking strategies. Despite the stacking method improves when true labels are used (STA$^{\boldsymbol{y}}$), the results are still worse than those of AID classifier. This different performance must come from the aggregating idea, given that the only difference between both algorithms is the way they decide the final predictions. The fact that AID prevails

Table 4. AID vs. state-of-the-art methods: Precision, Recall, F_1 and Jaccard index

Precision	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	48.19(3)	26.60(7)	28.92(6)	47.08(5)	47.69(4)	48.70(2)	**49.08** (1)
Emotions	56.36(4)	52.42(7)	**67.54**(1)	52.79(6)	56.26(5)	62.30(2)	59.03 (3)
Enron	**69.99**(1)	54.90(6)	52.75(7)	56.05(5)	65.19(4)	66.10(3)	66.37 (2)
Genbase	99.52(3.5)	97.70(7)	98.90(6)	**99.57**(1)	99.52(3.5)	99.52(3.5)	99.52 (3.5)
Image	44.23(7)	44.33(6)	48.52(2)	46.66(4)	45.99(5)	**53.97**(1)	48.18 (3)
Mediamill	78.81(2)	76.93(4)	73.52(5)	**80.40**(1)	77.87(3)	70.91(6)	70.16 (7)
Medical	78.94(5)	62.43(7)	63.40(6)	80.79(4)	81.04(2)	**82.50**(1)	80.88 (3)
Reuters	85.79(5)	82.23(6)	70.71(7)	**89.62**(1)	86.41(4)	87.53(2)	87.48 (3)
Scene	61.46(7)	69.71(3)	71.40(2)	69.69(4)	67.45(6)	67.81(5)	**71.89** (1)
Slashdot	46.06(4)	6.15(7)	8.09(6)	50.91(2)	42.79(5)	**53.32**(1)	46.98 (3)
Yeast	71.13(3)	**72.92**(1)	71.75(2)	68.62(5)	70.58(4)	66.50(7)	67.83 (6)
Avg. rank	(4.05)	(5.55)	(4.55)	(3.45)	(4.14)	(**3.05**)	(3.23)
Recall	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	33.80(4)	14.06(7)	21.50(6)	**41.97**(1)	33.78(5)	36.86(2)	36.47 (3)
Emotions	48.16(6)	37.73(7)	64.54(2)	57.19(3)	48.92(5)	**68.03**(1)	53.44 (4)
Enron	50.50(4)	37.04(7)	38.04(6)	54.07(3)	45.79(5)	**59.89**(1)	54.86 (2)
Genbase	99.07(4.5)	94.96(7)	99.14(2)	**99.57**(1)	99.07(4.5)	99.07(4.5)	99.07 (4.5)
Image	43.32(6)	39.11(7)	43.68(5)	49.73(2)	44.10(4)	**55.96**(1)	47.74 (3)
Mediamill	52.33(5)	53.78(4)	56.69(3)	49.57(7)	51.25(6)	**59.25**(1)	58.02 (2)
Medical	78.34(5)	59.01(7)	65.05(6)	81.00(2)	79.01(4)	**83.86**(1)	80.57 (3)
Reuters	84.90(5)	81.09(6)	69.45(7)	89.63(2)	85.19(4)	**91.90**(1)	87.90 (3)
Scene	62.87(7)	68.73(5)	69.75(3)	69.52(4)	66.43(6)	**87.55**(1)	77.10 (2)
Slashdot	44.21(4)	5.69(7)	7.67(6)	53.18(2)	39.93(5)	**71.42**(1)	45.05 (3)
Yeast	58.86(6)	56.89(7)	60.41(4)	61.84(3)	59.40(5)	**66.42**(1)	62.50 (2)
Avg. rank	(5.14)	(6.45)	(4.55)	(2.73)	(4.86)	(**1.41**)	(2.86)
$\boldsymbol{F_1}$	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	37.02(4)	16.98(7)	22.38(6)	**41.28**(1)	36.86(5)	39.22(2)	39.03 (3)
Emotions	49.20(6)	41.60(7)	**62.97**(1)	51.95(4)	49.81(5)	62.05(2)	53.31 (3)
Enron	55.66(3)	41.82(6)	41.52(7)	52.43(4)	51.14(5)	**59.97**(1)	56.73 (2)
Genbase	99.18(3.5)	95.81(7)	98.78(6)	**99.50**(1)	99.18(3.5)	99.18(3.5)	99.18 (3.5)
Image	42.12(6)	40.63(7)	44.91(4)	46.32(2)	43.46(5)	**52.69**(1)	46.18 (3)
Mediamill	59.17(5)	59.55(4)	60.17(2)	57.72(7)	58.15(6)	**60.39**(1)	59.67 (3)
Medical	77.33(5)	59.41(7)	62.19(6)	79.65(2)	78.83(4)	**81.66**(1)	79.35 (3)
Reuters	84.13(5)	80.50(6)	69.10(7)	**88.58**(1)	84.69(4)	87.74(2)	86.43 (3)
Scene	61.25(7)	68.49(5)	69.97(3)	68.84(4)	66.31(6)	71.38(2)	**72.85** (1)
Slashdot	44.33(4)	5.84(7)	7.73(6)	50.49(2)	40.66(5)	**55.97**(1)	45.18 (3)
Yeast	61.68(6)	60.97(7)	62.85(2)	62.48(3)	61.80(5)	**63.28**(1)	62.42 (4)
Avg. rank	(4.95)	(6.36)	(4.55)	(2.82)	(4.86)	(**1.59**)	(2.86)
Jaccard	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	31.50(4)	13.61(7)	18.09(6)	**34.28**(1)	31.46(5)	33.59(2)	33.28 (3)
Emotions	42.27(6)	34.09(7)	**55.08**(1)	42.98(5)	43.24(4)	52.89(2)	46.12 (3)
Enron	44.69(2)	31.83(7)	31.99(6)	41.30(4)	39.68(5)	**48.50**(1)	44.23 (3)
Genbase	98.94(3.5)	94.86(7)	98.25(6)	**99.29**(1)	98.94(3.5)	98.94(3.5)	98.94 (3.5)
Image	38.60(6)	38.45(7)	42.46(2)	42.15(4)	40.15(5)	**47.88**(1)	42.43 (3)
Mediamill	46.70(4)	48.11(3)	**48.82**(1)	45.10(6)	44.69(7)	48.13(2)	46.00 (5)
Medical	74.51(5)	56.76(7)	58.19(6)	76.88(2)	76.33(4)	**78.40**(1)	76.37 (3)
Reuters	81.67(5)	78.11(6)	67.10(7)	**86.38**(1)	82.41(4)	84.37(2)	83.80 (3)
Scene	59.41(7)	67.03(4)	68.77(2)	67.28(3)	65.05(6)	65.90(5)	**69.74** (1)
Slashdot	42.71(4)	5.68(7)	7.42(6)	47.18(2)	39.32(5)	**49.98**(1)	43.50 (3)
Yeast	50.71(6)	50.50(7)	**52.65**(1)	51.75(3)	50.96(5)	52.40(2)	51.22 (4)
Avg. rank	(4.77)	(6.27)	(4.00)	(2.91)	(4.86)	(**2.05**)	(3.14)

Table 5. AID vs. state-of-the-art methods: Hamming loss and 0/1 loss

Hamming	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID$^{\boldsymbol{p}}$
Bibtex	**1.21**(1.5)	1.36(5)	1.60(7)	1.49(6)	**1.21**(1.5)	1.22(3.5)	1.22 (3.5)
Emotions	22.03(4)	26.21(6)	**18.72**(1)	28.01(7)	21.72(3)	23.04(5)	21.47 (2)
Enron	**4.46**(1)	5.22(5)	5.60(6)	5.85(7)	4.72(3)	4.82(4)	4.69 (2)
Genbase	0.08(3.5)	0.45(7)	0.19(6)	**0.06**(1)	0.08(3.5)	0.08(3.5)	0.08 (3.5)
Image	20.25(4.5)	19.28(2)	**18.75**(1)	24.40(7)	19.80(3)	21.90(6)	20.25 (4.5)
Mediamill	2.76(2)	**2.70**(1)	2.82(4)	2.81(3)	2.86(5)	3.03(7)	2.99 (6)
Medical	0.99(5)	1.56(6)	1.90(7)	**0.95**(2)	**0.95**(2)	**0.95**(2)	0.97 (4)
Reuters	4.58(4)	6.03(6)	8.32(7)	**3.85**(1)	4.42(2)	5.81(5)	4.44 (3)
Scene	9.83(4)	8.66(2)	**8.38**(1)	9.90(5)	9.06(3)	18.66(7)	10.28 (6)
Slashdot	**3.73**(1)	5.18(6)	5.17(5)	4.65(4)	3.78(3)	8.81(7)	3.75 (2)
Yeast	19.81(3)	19.43(2)	**19.18**(1)	20.30(5)	19.98(4)	21.29(7)	20.66 (6)
Avg. rank	(3.05)	(4.36)	(4.18)	(4.36)	(**3.00**)	(5.18)	(3.86)
0/1 loss	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID$^{\boldsymbol{p}}$
Bibtex	82.83(4)	94.06(7)	91.64(6)	83.56(5)	82.61(3)	**81.49**(1)	81.96 (2)
Emotions	79.42(5)	87.00(7)	**68.97**(1)	83.45(6)	77.05(4)	74.19(2)	76.05 (3)
Enron	86.90(2)	94.89(7)	92.30(4)	88.49(3)	93.07(6)	**85.37**(1)	92.54 (5)
Genbase	1.81(3.5)	8.16(7)	4.08(6)	**1.51**(1)	1.81(3.5)	1.81(3.5)	1.81 (3.5)
Image	71.50(7)	67.95(3)	**64.70**(1)	69.70(6)	69.40(5)	65.75(2)	68.35 (4)
Mediamill	90.36(4)	86.24(2)	**85.86**(1)	91.14(5)	93.80(6)	88.10(3)	94.48 (7)
Medical	33.94(5)	51.12(6)	52.98(7)	31.39(3)	31.19(2)	**31.18**(1)	32.62 (4)
Reuters	25.69(5)	29.04(6)	38.88(7)	**20.24**(1)	24.43(3)	25.16(4)	24.10 (2)
Scene	46.03(6)	37.35(2.5)	**34.82**(1)	37.35(2.5)	38.72(4)	46.07(7)	39.26 (5)
Slashdot	61.95(2)	94.76(7)	93.47(6)	62.11(3)	64.57(5)	63.20(4)	**61.34** (1)
Yeast	84.53(6)	82.29(2)	**79.19**(1)	83.08(3.5)	83.58(5)	83.08(3.5)	85.85 (7)
Avg. rank	(4.50)	(5.14)	(3.73)	(3.55)	(4.23)	(**2.91**)	(3.95)

in all measures, although the differences are sometimes small (Hamming and 0/1 loss), suggests that the aggregation idea helps to increase the performance. We observe even bigger differences when comparing the scores of BR and AID. These facts corroborate our original assumption of the existence of two kind of labels: some labels are explained by the mere description of the examples, whereas others have been assigned as a consequence of other labels. The differences in the performance may provide an estimation about the proportion of existing labels of each kind and the degree of overlapping.

4.2 AID Classifier vs. State-of-the-Art Methods

The second group of experiments was designed to compare our approach AID with other well-known multi-label classifiers: BR, MLkNN, RAkEL, IBLR and ECC*. We also included AID$^{\boldsymbol{p}}$ because of its good performance in all measures, specially in terms of Hamming loss. Table 4 and Table 5 show all the scores.

As it can be seen, the three top positions in the ranking are occupied by AID RAkEL and AID$^{\boldsymbol{p}}$ for all measures, except for Hamming loss, for which ECC*, BR and also AID$^{\boldsymbol{p}}$ win the others. It is remarkable than MLkNN is always placed in the last position for all measures. The Bergmann-Hommel's test (see Table 6) shows that AID is significantly better than BR, MLkNN, IBLR and ECC* in Recall, F_1 and Jaccard index, and there are not significant differences in Precision, Hamming and 0/1 losses. Also, AID$^{\boldsymbol{p}}$ presents significant differences

Table 6. Pairs of methods with significant differences according to Bergmann-Hommel's test. The number 90% or 95% indicates the significant level.

		Recall	F_1	Jaccard	Hamming			Recall	F_1	Jaccard
AID	$\succ$ BR	95%	95%	95%	-90%	AID	$\succ$ BR	95%	95%	95%
AID	$\succ$ STA	95%	95%	95%		AID	$\succ$ MLkNN	95%	95%	95%
AID	$\succ$ STA$^{\boldsymbol{p}}$	95%	95%	95%		AID	$\succ$ IBLR	95%	95%	
AID	$\succ$ AID$^{\boldsymbol{y}'}$		95%	95%		AID	$\succ$ ECC*	95%	95%	95%
AID$^{\boldsymbol{p}}$	$\succ$ BR	95%	95%	90%		AID$^{\boldsymbol{p}}$	$\succ$ MLkNN	95%	95%	95%
AID$^{\boldsymbol{p}}$	$\succ$ STA$^{\boldsymbol{p}}$	90%				AID$^{\boldsymbol{p}}$	$\succ$ BR	90%		
AID$^{\boldsymbol{y}'}$	$\succ$ BR	95%				RAkEL	$\succ$ MLkNN	95%	95%	95%
AID$^{\boldsymbol{y}'}$	$\succ$ STA$^{\boldsymbol{p}}$	90%				RAkEL	$\succ$ BR	90%		
STA$^{\boldsymbol{y}}$	$\succ$ BR	95%	95%	90%	-90%					
STA$^{\boldsymbol{y}}$	$\succ$ STA$^{\boldsymbol{p}}$	95%								

in those measures but only with regard to BR and MLkNN. RAkEL is also quite competitive, since it is significantly better than BR in Recall and than MLkNN in Recall, F_1 and Jaccard index.

From the results obtained, one can extract that AID$^{\boldsymbol{p}}$ keeps a steady behavior over all performance measures, whereas AID improves in Precision, Recall, F_1, Jaccard index and 0/1 loss and ECC* does it in Hamming loss. Both seem to be opposing methods with regard to the behavior of the measures, whereas RAkEL seems to be placed in between. Hence, if there is not a clear target measure to optimize, then AID$^{\boldsymbol{p}}$ may be a good choice, followed by RAkEL. Conversely, if the goal is to maximize Hamming loss, then ECC*, or even BR must be chosen. Finally, for maximizing the rest of measures AID is more promising.

5 Conclusions

This paper proposes a multi-label learning approach, called AID, whose main property is to aggregate independent and dependent models. Under the assumption that, for a given instance, certain labels can be predicted using only the description of the object, but others require to consider their dependences with respect to other labels, the idea is to combine classifiers aimed to learn each of these kind of relationships. In this work, we study the properties of our approach in the context of multi-label classification, but our framework is flexible enough to be adapted to other learning tasks, specially to multi-label ranking, and it can also be extended in some directions.

Several experiments over a benchmark multi-label data sets were carried out using different learning approaches. None of the methods outperforms the others for all measures considered, but AID classifier exhibits a very competitive performance, specially in terms of Recall, F_1 and Jaccard index, with regard to other state-of-the-art algorithms previously proposed in the literature. Besides, the computational complexity of AID is linear with respect to the number of labels.

References

1. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Machine Learning 76(2-3), 211–225 (2009)
2. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: European Conf. on Data Mining and Knowledge Discovery, pp. 42–53 (2001)
3. Dembczynski, K., Cheng, W., Hüllermeier, E.: Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains. In: ICML, pp. 279–286 (2010)
4. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: Regret analysis for performance metrics in multi-label classification: The case of hamming and subset zero-one loss. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6321, pp. 280–295. Springer, Heidelberg (2010)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
6. Elisseeff, A., Weston, J.: A Kernel Method for Multi-Labelled Classification. In: ACM Conf. on Research and Develop. in Infor. Retrieval, pp. 274–281 (2005)
7. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Machine Learning 73, 133–153 (2008)
8. García, S., Herrera, F.: An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008)
9. Ghamrawi, N., McCallum, A.: Collective multi-label classification. In: ACM Int. Conf. on Information and Knowledge Management, pp. 195–200. ACM, New York (2005)
10. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Pacific-Asia Conf. on Know. Disc. and Data Mining, pp. 22–30 (2004)
11. Lin, C.-J., Weng, R.C., Keerthi, S.S.: Trust region Newton method for logistic regression. Journal of Machine Learning Research 9(apr), 627–650 (2008)
12. McCallum, A.K.: Multi-label text classification with a mixture model trained by em. In: AAAI 1999 Workshop on Text Learning (1999)
13. Qi, G.J., Hua, X.S., Rui, Y., Tang, J., Mei, T., Zhang, H.J.: Correlative multi-label video annotation. In: Proceedings of the International conference on Multimedia, pp. 17–26. ACM, New York (2007)
14. Read, J., Pfahringer, B., Holmes, G.: Multi-label classification using ensembles of pruned sets. In: IEEE Int. Conf. on Data Mining, pp. 995–1000. IEEE, Los Alamitos (2008)
15. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 254–269. Springer, Heidelberg (2009)
16. Schapire, R.E., Singer, Y.: Boostexter: A boosting-based system for text categorization. Machine Learning, 135–168 (2000)
17. Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., Vlahavas, I.: Correlation-based pruning of stacked binary relevance models for multi-label learning. In: Workshop on Learning from Multi-Label Data, Bled, Slovenia, pp. 101–116 (2009)
18. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook, pp. 667–685 (2010)
19. Tsoumakas, G., Vlahavas, I.P.: Random k-Labelsets: An Ensemble Method for Multilabel Classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007)

20. Wolpert, D.H.: Stacked generalization. Neural Networks 5, 214–259 (1992)
21. Zhang, M.-L., Zhou, Z.-H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. on Knowl. and Data Eng. 18, 1338–1351 (2006)
22. Zhang, M.-L., Zhou, Z.-H.: Ml-knn: A lazy learning approach to multi-label learning. Pattern Recognition 40(7), 2038–2048 (2007)

Tensor Factorization Using Auxiliary Information

Atsuhiro Narita[1], Kohei Hayashi[2], Ryota Tomioka[1], and Hisashi Kashima[1,3]

[1] Department of Mathematical Informatics,
The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{atsuhiro_narita,tomioka,kashima}@mist.i.u-tokyo.ac.jp
[2] Graduate School of Information Science,
Nara Institute of Science and Technology,
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
kohei-h@is.naist.jp
[3] Basic Research Programs PRESTO,
Synthesis of Knowledge for Information Oriented Society

Abstract. Most of the existing analysis methods for tensors (or multi-way arrays) only assume that tensors to be completed are of low rank. However, for example, when they are applied to tensor completion problems, their prediction accuracy tends to be significantly worse when only limited entries are observed. In this paper, we propose to use relationships among data as auxiliary information in addition to the low-rank assumption to improve the quality of tensor decomposition. We introduce two regularization approaches using graph Laplacians induced from the relationships, and design iterative algorithms for approximate solutions. Numerical experiments on tensor completion using synthetic and benchmark datasets show that the use of auxiliary information improves completion accuracy over the existing methods based only on the low-rank assumption, especially when observations are sparse.

1 Introduction

In real data analysis applications, we often have to face handling multi-object relationships. For example, in on-line marketing scenarios, we analyze relationships among customers, items, and time to capture temporal dynamics of customers' interests and utilize them for recommendation. In social network analysis, interactions among people and their interaction types are the focus of interest. Similar situations arise in bio- and chemo-informatics as protein-protein interactions and drug-target interactions under various conditions. Tensors (or multi-way arrays) [10] are highly suitable representation for such multi-object relationships (Fig. 1). Tensor analysis methods, especially, models and efficient algorithms for low-rank tensor decompositions have been extensively studied and applied to many real-world problems. CANDECOMP/PARAFAC(CP)-decomposition and Tucker decomposition are two widely-used low rank decompositions of tensors (Fig. 2).

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 501–516, 2011.

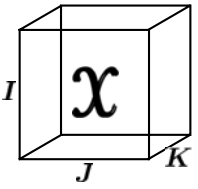
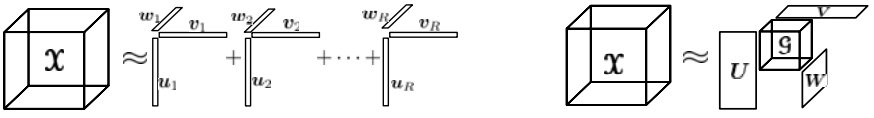


Fig. 1. A third-order tensor (or, a three-way array) $\mathcal{X}$ of size $I \times J \times K$ repesents relationships among three sets of objects, S_1, S_2 and S_3, each of which size is I, J, and K, respectively



(a) CP decomposition. (b) Tucker decomposition.

Fig. 2. Two widely used low-rank tensor decompositions: CP-decomposition and Tucker decomposition

Tensor completion is one of important applications of tensor analysis methods. Given a tensor with some of its elements being missing, the task is to impute the missing values. In the context of the previous on-line marketing scenario, given the observations for some (customer, item, time)-tuples, we can make recommendations by imputing unobserved combinations of them. Tensor completion is also used for link prediction [6,9] and tag recommendation [15]. Similar to the other tensor analysis methods, low-rank assumption of tensors is often used for imputing missing values. However, when observations are sparse, in other words, the fraction of unobserved elements is high, predictive accuracy of tensor completion methods only with the low-rank assumption tends to be worse. For example, Figure 3 shows the prediction errors by CP-decomposition against the fraction of unobserved elements for a particular dataset. (The experimental setting is the same as that for Fig. 4(c), which will be described in detail in the experimental section.) We can see the accuracy of tensor completion severely degrades when observations are sparse. This fact implies that the low-rank assumption by itself is not sufficient and we need other assumptions to introduce more prior knowledge of subjects.

In many cases, we have not only relational information among objects, but also information on the objects themselves. For example, in the on-line marketing scenarios, each customer has his/her demographic information, and each item has its product information. We consider exploiting these auxiliary information for improving the prediction accuracy of tensor decomposition, especially for sparse cases. Inspired by the work by Li *et al.* [12] which incorporates object similarity into matrix factorization, we exploit the auxiliary information given as similarity matrices in a regularization framework for tensor factorization. We propose two specific regularization methods, one of which we call "within-mode

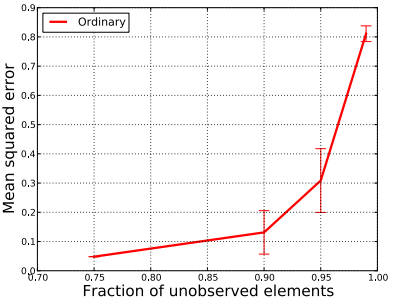


Fig. 3. The tensor completion performance by CP-decomposition for the 'Flow injection' dataset. Prediction accuracy severely degenerates when observations are sparse.

regularization" is a natural extension of the method proposed by Li *et al.* for matrix factorization. It uses the graph Laplacians induced by the similarity matrices to force two similar objects in each mode to behave similarly, in other words, to have similar factors. The second method we call "cross-mode regularization" exploits the similarity information more aggressively to address extremely sparse cases. We apply the two proposed regularization methods to each of CP-decomposition and Tucker decomposition, and give iterative decomposition algorithms for obtaining approximate solutions. In each iteration, we solve a particular Sylvester equation for CP-decomposition, or obtain eigendecomposition for Tucker decomposition. To best of our knowledge, our work is the first to incorporates auxiliary information into tensor decomposition.

Finally, we show experimental results on missing value imputation using both synthetic and real benchmark datasets. We test two kinds of assumptions on missing elements. The first one is element-wise missing where each element is missing independently. The second one is slice-wise missing (in other words, object-wise missing), where missing values occur in a more bursty manner, and all of the elements related to some objects are completely missing. The experimental results demonstrate that the use of auxiliary information improves imputation accuracy when observations are sparse, and the cross-mode regularization method especially works well in extremely sparse slice-wise missing cases.

The rest of the paper is organized as follows. Section 2 reviews the existing low-rank tensor decomposition methods, and introduces the tensor completion problem with auxiliary information that we focus on in this paper. In Section 3, we propose two regularization strategies, within-mode regularization and cross-mode regularization, for incorporating auxiliary information in tensor decomposition, and give their approximate solutions. Section 4 shows the experimental results using several datasets to demonstrate the proposed methods work well especially when observations are sparse. Section 5 reviews related work, and Section 6 concludes this paper with some ideas for future directions.

2 Tensor Completion Problem with Auxiliary Information

We first review the existing low-rank tensor decomposition methods, and then formulate the tensor completion problem with auxiliary information.

2.1 Tensor Analysis Using Low-Rank Decomposition

Let $\boldsymbol{\mathcal{X}}$ be third-order tensor (i.e. a three-way array) with $I \times J \times K$ real-valued elements[1]. The third-order tensor $\boldsymbol{\mathcal{X}}$ models relationships among objects from three sets S_1, S_2, and S_3. For example, in the context of on-line marketing, S_1, S_2, and S_3 represent sets of customers, items, and time stamps, respectively. The (i, j, k)-th element $[\boldsymbol{\mathcal{X}}]_{i,j,k}$ indicates the i-th user's rating of the j-th item at time k.

We often assume the tensor is of "low-rank" when we analyze tensor-represented data. In contrast to matrices (that are special cases of tensors), definitions of the "low-rank" tensor are not unique. CANDECOMP/PARAFAC (CP)-decomposition and Tucker decomposition are often used as definitions of low-rank tensors.

The CP-decomposition is a natural extension of the matrix rank, and it approximates a tensor by the sum of R rank-1 tensors. The CP-decomposition $\hat{\boldsymbol{\mathcal{X}}}$ of $\boldsymbol{\mathcal{X}}$ is defined as

$$\hat{\boldsymbol{\mathcal{X}}} \equiv \sum_{i=1}^{R} \mathbf{u}_i \circ \mathbf{v}_i \circ \mathbf{w}_i,$$

where $\circ$ indicates the outer product operation. Or, it can also be represented by using mode-i multiplications as

$$\hat{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{J}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}, \tag{1}$$

where $\boldsymbol{\mathcal{J}} \in \mathbb{R}^{R \times R \times R}$ is a unit tensor with all of its super-diagonal elements being 1 and the other elements being 0, $\mathbf{U} \in \mathbb{R}^{I \times R}$, $\mathbf{V} \in \mathbb{R}^{J \times R}$, $\mathbf{W} \in \mathbb{R}^{K \times R}$ are factor matrices, and $\times_i$ is the mode-i multiplication [10]. When the left-hand side is equal to the right-hand side in the above relation, we say $\boldsymbol{\mathcal{X}}$ is of rank R.

The Tucker decomposition approximates a tensor with a small "core tensor" and factor matrices, which is defined as

$$\hat{\boldsymbol{\mathcal{X}}} \equiv \boldsymbol{\mathcal{G}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}, \tag{2}$$

where $\boldsymbol{\mathcal{G}}$ is a (P, Q, R)-tensor and $\mathbf{U} \in \mathbb{R}^{I \times P}$, $\mathbf{V} \in \mathbb{R}^{J \times Q}$, $\mathbf{W} \in \mathbb{R}^{K \times R}$ are factor matrices. In this case, we say $\boldsymbol{\mathcal{X}}$ is of rank (P, Q, R).

For most of realistic case, observations are perturbed by noise, and the strict low-rank decompositions do not hold even when the "true" $\boldsymbol{\mathcal{X}}$ is actually of low-rank. Therefore, we try to find a decomposition $\hat{\boldsymbol{\mathcal{X}}}$ that best approximates

[1] For simplicity, we focus on third-order tensors in this paper. However, the discussion can be directly applied to higher-order tensors.

the original tensor $\boldsymbol{\mathcal{X}}$ in terms of the squared loss by the following optimization problem,

$$\text{minimize}_{\hat{\boldsymbol{\mathcal{X}}}} \quad \|\boldsymbol{\mathcal{X}} - \hat{\boldsymbol{\mathcal{X}}}\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ indicates the Frobenius norm, and $\hat{\boldsymbol{\mathcal{X}}}$ is defined by Eq. (1) for CP-decomposition, or by Eq. (2) for Tucker decomposition. It is generally hard to obtain the optimal solution, so we use approximation methods which optimize $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ alternately.

2.2 Tensor Completion with Auxiliary Information

Among various tensor-related problems, tensor completion problem is one of the important problems, where the task is to impute the missing values of a given tensor with missing values. The low-rank assumption is usually used as a heuristic for inferring the missing parts. Since not all of the elements are observed in the optimization problem (3) in this case, the EM algorithm is often applied to this purpose [18,20]. First, we fill the missing parts with some initial estimates (such as the average of the observed elements), and apply tensor decomposition to the filled tensor. We then obtain new estimates by assembling the decomposed tensor. We continue the decomposition step and the reconstruction step until convergence to obtain final estimates.

Since the EM algorithm uses unobserved elements for its computation, it is not efficient enough for large-scale data. Therefore, another approach modifies the objective function (3) to focus only on observed parts [1]. In this paper, we construct our methods based on the EM-based approach, however, the basic idea can also be applied to the other approaches.

The low-rank assumption of tensors makes it possible to impute missing values. However, when observations are sparse, in other words, the fraction of unobserved elements is high, predictive accuracy of tensor completion methods only with the low-rank assumption severely degrades. (See Figure 3 showing the predictive errors against the fraction of unobserved elements for a dataset.) Therefore, the low-rank assumption by itself is not sufficient, and we need other assumptions for obtaining satisfactory prediction accuracy.

In many realistic cases, we have not only relational information represented as tensors, but also information on the objects forming the relationships. For example, in the (customer, item, time)-relationships, each customer has his/her demographic information, and each item has its product information. We also know that time is continuous and can assume temporal smoothness. Therefore, we assume that we have similarity measures for S_1, S_2, and S_3, each of which corresponds to the sets of objects for each of the three modes. We define a non-negative symmetric matrix $\mathbf{A}_1$ for representing the similarity between arbitrary two objects in S_1. $\mathbf{A}_2$ and $\mathbf{A}_3$ are defined similarly.

We consider exploiting these auxiliary information for improving the prediction accuracy by tensor decomposition, especially for sparse observations. The tensor completion problem that we focus on in this paper is summarized as follows.

Problem: (Third-order) tensor completion with auxiliary information

- **INPUT**
  - A third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, some of whose elements are observed and the others are unobserved.
  - Three non-negative symmetric similarity matrices $\mathbf{A}_1 \in \mathbb{R}^{+I \times I}$, $\mathbf{A}_2 \in \mathbb{R}^{+J \times J}$, and $\mathbf{A}_3 \in \mathbb{R}^{+K \times K}$, each of which corresponds to one of the three modes of $\boldsymbol{\mathcal{X}}$.
- **OUTPUT:** A decomposition $\hat{\boldsymbol{\mathcal{X}}}$ defined by either Eq. (1) for CP-decomposition or Eq. (2) for Tucker decomposition.

3 Proposed Methods: Within-Mode and Cross-Mode Regularization

In this section, we propose two regularization methods for incorporating auxiliary information into tensor factorization. Both CP-decomposition and Tucker decomposition are generalized with the regularization framework.

3.1 Regularization Using Auxiliary Similarity Matrices

Given three object similarity matrices $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$ besides $\boldsymbol{\mathcal{X}}$, how can we use them for improving tensor decompositions? Probably, one of natural assumptions we can make is that "two similar objects behave similarly". We implement this idea as regularization terms for the optimization problem (3). Namely, instead of the objective function in Eq. (3), we minimize the following regularized objective function with a regularization term $R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$.

$$f(\hat{\boldsymbol{\mathcal{X}}}) \equiv \frac{1}{2}\|\boldsymbol{\mathcal{X}} - \hat{\boldsymbol{\mathcal{X}}}\|_F^2 + \frac{\alpha}{2} R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3). \tag{4}$$

In Eq. (4), α is a positive regularization constant.

We propose two specific choices of the regularization term $R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$. The first method we call "within-mode regularization" is a natural extension of the method proposed by Li *et al.* [12] for matrix factorization. It uses the graph Laplacians induced by the three similarity matrices to force two similar objects in each mode to behave similarly, in other words, to have similar factors. The second method we call "cross-mode regularization" exploits the similarity information more aggressively to address extremely sparse cases. It uses the graph Laplacian induced by the Kronecker product of the three similarity matrices to regularize factors for all of the modes at the same time by taking interactions across different modes into account.

3.2 Method 1: Within-Mode Regularization

The first regularization term we propose regularizes factor matrices for each mode using the similarity matrices. The "within-mode" regularization term is defined as

$$R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \equiv \mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right), \tag{5}$$

where $\mathbf{L}_1$ is the Laplacian matrix induced from the similarity matrix $\mathbf{A}_1$ for the object set S_1. The Laplacian matrix is defined as

$$\mathbf{L}_1 \equiv \mathbf{D}_1 - \mathbf{A}_1,$$

where $\mathbf{D}_1$ is the diagonal matrix whose i-th diagonal element is the sum of all of the elements in the i-th row of $\mathbf{A}_1$. The Laplacian matrices for the other two sets, $\mathbf{L}_2 \equiv \mathbf{D}_2 - \mathbf{A}_2$ and $\mathbf{L}_3 \equiv \mathbf{D}_3 - \mathbf{A}_3$, are defined similarly.

To interpret the regularization term, we note $\mathrm{tr}(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U})$ can be rewritten as

$$\mathrm{tr}(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}) = \sum_{i,j=1}^{I} [\mathbf{A}_1]_{i,j} \sum_{r=1}^{R} \left([\mathbf{U}]_{i,r} - [\mathbf{U}]_{j,r}\right)^2, \tag{6}$$

where $[\cdot]_{i,j}$ denotes the (i,j)-th element of a matrix. This term implies that, if two objects (say, $s_i, s_j \in S_1$) are similar to each other (that is, $[\mathbf{A}_1]_{i,j}$ is large), the corresponding factor vectors ($[\mathbf{U}]_{i*}$ and $[\mathbf{U}]_{j*}$) should be similar to each other.

CP-decomposition. The objective function for CP-decomposition with the within-mode regularization is written as

$$\begin{aligned} f(\mathcal{G}, \mathbf{U}, \mathbf{V}, \mathbf{W}) \equiv & \frac{1}{2} \|\mathcal{X} - \mathcal{J} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 \\ & + \frac{\alpha}{2} \mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right). \end{aligned} \tag{7}$$

Eq. (7) is not a convex function for $(\mathbf{U}, \mathbf{V}, \mathbf{W})$, but is convex for each of $\mathbf{U}, \mathbf{V}$, and $\mathbf{W}$. Therefore, we optimize one of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ with fixing the others to the current values, and alternately update them by changing the factor matrix to optimize.

Suppose we want to optimize $\mathbf{U}$ with fixing $\mathbf{V}$ and $\mathbf{W}$. Unfolding Eq. (7) by the first mode (i.e. making the mode-1 matricization), we obtain

$$\begin{aligned} f(\mathcal{G}, \mathbf{U}, \mathbf{V}, \mathbf{W}) = & \frac{1}{2} \|\mathbf{X}_{(1)} - \mathbf{U} (\mathbf{W} \odot \mathbf{V})^\top\|_F^2 \\ & + \frac{\alpha}{2} \mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right) \\ = & \frac{1}{2} \mathrm{tr}\left(\left(\mathbf{X}_{(1)} - \mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top\right)^\top \left(\mathbf{X}_{(1)} - \mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top\right)\right) \\ & + \frac{\alpha}{2} \mathrm{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right), \end{aligned}$$

where $\mathbf{X}_{(n)}$ denotes the mode-n matricization of $\mathcal{X}$, and $\odot$ denotes the Khatri-Rao product [10]. Differentiating this with respect to $\mathbf{U}$, and setting it to be zero gives the Sylvester equation,

$$\mathbf{U}(\mathbf{W} \odot \mathbf{V})^\top (\mathbf{W} \odot \mathbf{V}) + \alpha \mathbf{L}_1 \mathbf{U} = \mathbf{U}\left(\mathbf{V}^\top \mathbf{V} * \mathbf{W}^\top \mathbf{W}\right) + \alpha \mathbf{L}_1 \mathbf{U} = \mathbf{X}_{(1)} (\mathbf{W} \odot \mathbf{V}),$$

where $*$ denotes the Hadamard product (i.e. element-wise product). The Sylvester equation can be solved by several numerical approaches such as the one implemented as the `dlyap` function in MATLAB®.

Tucker Decomposition. In the case of Tucker decomposition, the objective function becomes

$$f(\mathbf{U},\mathbf{V},\mathbf{W}) \equiv \|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 + \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U} + \mathbf{V}^\top \mathbf{L}_2 \mathbf{V} + \mathbf{W}^\top \mathbf{L}_3 \mathbf{W}\right). \tag{8}$$

We minimize the objective function (8) under the orthogonality constraints, $\mathbf{U}^\top\mathbf{U} = \mathbf{I}, \mathbf{V}^\top\mathbf{V} = \mathbf{I}$, and $\mathbf{W}^\top\mathbf{W} = \mathbf{I}$. Noting the core tensor $\boldsymbol{\mathcal{G}}$ is obtained as the closed form solution,

$$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top,$$

the first term of Eq. (8) can be rewritten as

$$\begin{aligned}\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 &= \|\boldsymbol{\mathcal{X}}\|_F^2 - \|\boldsymbol{\mathcal{G}}\|_F^2 \\ &= \|\boldsymbol{\mathcal{X}}\|_F^2 - \|\boldsymbol{\mathcal{X}} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top\|_F^2.\end{aligned}$$

When we optimize Eq. (8) with respect to $\mathbf{U}$, by ignoring the terms unrelated to $\mathbf{U}$, we obtain an equivalent maximization problem of

$$\tilde{f}(\mathbf{U}) \equiv \|\boldsymbol{\mathcal{X}} \times_1 \mathbf{U}^\top \times_2 \mathbf{V}^\top \times_3 \mathbf{W}^\top\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right). \tag{9}$$

Unfolding Eq. (9) by the first mode, we have

$$\tilde{f}(\mathbf{U}) = \|\mathbf{U}^\top \mathbf{X}_{(1)} (\mathbf{W} \otimes \mathbf{V})\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right).$$

Setting $\mathbf{S} \equiv \mathbf{X}_{(1)} (\mathbf{W} \otimes \mathbf{V}) = (\boldsymbol{\mathcal{X}} \times_2 \mathbf{V} \times_3 \mathbf{W})_{(1)}$, $\tilde{f}(\mathbf{U})$ is further rewritten as

$$\tilde{f}(\mathbf{U}) = \|\mathbf{U}^\top \mathbf{S}\|_F^2 - \alpha \operatorname{tr}\left(\mathbf{U}^\top \mathbf{L}_1 \mathbf{U}\right) = \operatorname{tr}\left(\mathbf{U}^\top \left(\mathbf{S}\mathbf{S}^\top - \alpha \mathbf{L}_1\right) \mathbf{U}\right). \tag{10}$$

The maximizer of Eq. (10) satisfying the orthogonality constraint $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ is obtained as the I leading eigenvectors of $\mathbf{S}\mathbf{S}^\top - \alpha\mathbf{L}_1$.

3.3 Proposed Method 2: Cross-Mode Regularization

The within-mode regularization scheme regularizes the elements only inside each factor matrix, because each element of $\mathbf{U}$ interacts only with at most $I - 1$ elements within $\mathbf{U}$. This fact sometimes limits the effect of the regularization when we have bursty missing values, For example, slice-level missing situations where no observations are given for some objects often occurs in the context of recommender systems as the "cold-start" problem. In such cases, the within-mode regularization can be sometimes too conservative.

The second regularization function we propose exploits the given auxiliary information more aggressively. It combines the given similarity matrices to co-regularize combinations of elements across different modes as

$$R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) \equiv \operatorname{tr}\left((\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U})^\top \mathbf{L} (\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U})\right), \tag{11}$$

where the $IJK \times IJK$-Laplacian matrix $\mathbf{L}$ is defined as

$$\mathbf{L} \equiv \mathbf{D}_3 \otimes \mathbf{D}_2 \otimes \mathbf{D}_1 - \mathbf{A}_3 \otimes \mathbf{A}_2 \otimes \mathbf{A}_1.$$

The regularization term Eq. (11) is rewritten with the matrix elements as

$$R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) = \sum_{i,j,k=1}^{I,J,K} \sum_{\ell,m,n=1}^{I,J,K} [\mathbf{A}_1]_{i,\ell} [\mathbf{A}_2]_{j,m} [\mathbf{A}_3]_{k,n} \sum_{p,q,r=1}^{P,Q,R} \left([\mathbf{U}]_{i,p} [\mathbf{V}]_{j,q} [\mathbf{W}]_{k,r} - [\mathbf{U}]_{\ell,p} [\mathbf{V}]_{m,q} [\mathbf{W}]_{n,r} \right)^2,$$

which regularizes the combinations of elements from three different factors in contrast with the within-mode regularization (6) considering each mode independently.

The cross-mode regularization (11) can be seen as a natural variant of the within-mode regularization (5). Because, if we use the Kronecker sum $\oplus$ instead of the Kronecker product $\otimes$ in Eq. (11), it is reduced to Eq. (5) under the orthogonality constraints.

CP-decomposition. The objective function for the cross-mode regularized CP-decomposition is defined as

$$f(\mathbf{U}, \mathbf{V}, \mathbf{W}) \equiv \frac{1}{2} \| \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{J}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W} \|_F^2 + \frac{\alpha}{2} \mathrm{tr} \left((\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U})^\top \mathbf{L} (\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U}) \right).$$

Noting that Eq. (11) is simplified as

$$R(\hat{\boldsymbol{\mathcal{X}}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3) = \mathrm{tr} \left(\mathbf{W}^\top \mathbf{D}_3 \mathbf{W} \right) \mathrm{tr} \left(\mathbf{V}^\top \mathbf{D}_2 \mathbf{V} \right) \mathrm{tr} \left(\mathbf{U}^\top \mathbf{D}_1 \mathbf{U} \right) - \mathrm{tr} \left(\mathbf{W}^\top \mathbf{A}_3 \mathbf{W} \right) \mathrm{tr} \left(\mathbf{V}^\top \mathbf{A}_2 \mathbf{V} \right) \mathrm{tr} \left(\mathbf{U}^\top \mathbf{A}_1 \mathbf{U} \right),$$

similar to the within-mode regularization, we obtain the Sylvester equation for $\mathbf{U}$ as

$$\mathbf{U} (\mathbf{W} \odot \mathbf{V})^\top (\mathbf{W} \odot \mathbf{V}) + (D_{VW} \mathbf{D}_1 - A_{VW} \mathbf{A}_1) \mathbf{U} = \mathbf{X}_{(1)} (\mathbf{W} \odot \mathbf{V}),$$

where D_{VW} and A_{VW} are defined as follows.

$$D_{VW} \equiv \mathrm{tr} \left(\mathbf{W}^\top \mathbf{D}_3 \mathbf{W} \right) \mathrm{tr} \left(\mathbf{V}^\top \mathbf{D}_2 \mathbf{V} \right)$$
$$A_{VW} \equiv \mathrm{tr} \left(\mathbf{W}^\top \mathbf{A}_3 \mathbf{W} \right) \mathrm{tr} \left(\mathbf{V}^\top \mathbf{A}_2 \mathbf{V} \right)$$

Tucker Decomposition. The objective function for the cross-mode regularized Tucker decomposition is defined as

$$f(\boldsymbol{\mathcal{G}}, \mathbf{U}, \mathbf{V}, \mathbf{W}) \equiv \| \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W} \|_F^2 + \alpha \ \mathrm{tr} \left((\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U})^\top \mathbf{L} (\mathbf{W} \otimes \mathbf{V} \otimes \mathbf{U}) \right).$$

Again, we alternately minimize the objective function with respect to one of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$. By a similar derivation to that for the within-mode regularization, the optimal $\mathbf{U}$ with $\mathbf{V}$ and $\mathbf{W}$ fixed is obtained as the I leading eigenvectors of

$$\mathbf{S}\mathbf{S}^\top \mathbf{A} - \alpha (D_{VW} \mathbf{D}_1 - A_{VW} \mathbf{A}_1).$$

4 Experiments

We show some results of numerical experiments of third-order tensor completion problems using synthetic and real benchmark datasets, and demonstrate that introducing auxiliary information improves predictive accuracy especially when observations are sparse.

4.1 Datasets

Synthetic Dataset. The first dataset is synthetic tensors with correlated objects. We generate CP-decomposed tensors with $\mathbf{U} \in \mathbb{R}^{I \times R}, \mathbf{V} \in \mathbb{R}^{J \times R}$ and $\mathbf{W} \in \mathbb{R}^{K \times R}$ with the rank $R \equiv 2$ and $I \equiv J \equiv K \equiv 30$ by using the linear formulae,

$$
\begin{aligned}
[\mathbf{U}]_{ir} &\equiv i\epsilon_r + \epsilon'_r \ (1 \leq i \leq I, 1 \leq r \leq R) \\
[\mathbf{V}]_{jr} &\equiv j\zeta_r + \zeta'_r \ (1 \leq j \leq J, 1 \leq r \leq R) \\
[\mathbf{W}]_{kr} &\equiv k\eta_r + \eta'_r \ (1 \leq k \leq K, 1 \leq r \leq R),
\end{aligned}
$$

where $\{\epsilon_r, \epsilon'_r, \zeta_r, \zeta'_r, \eta_r, \eta'_r\}_{r=1}^{R}$ are constants generated by using the standard Gaussian distribution. A synthetic tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ is defined as

$$
\boldsymbol{\mathcal{X}} \equiv \boldsymbol{\mathcal{T}} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}.
$$

Since the columns of each factor matrix are generated by linear functions, the consecutive rows are similar to each other. Therefore, the similarity matrix for the i-th mode is naturally defined as the following tri-diagonal matrix.

$$
\mathbf{A}_i \equiv \begin{bmatrix} 0 & 1 & 0 & \cdots \\ 1 & 0 & 1 & \cdots \\ 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}
$$

Benchmark Dataset 1: Flow Injection. As a real benchmark dataset with auxiliary information, we used the 'Rank-deficient spectral FIA dataset'[2], which consists of results of flow injection analysis on 12 different chemical substances. They are represented as a tensor of size 12 (substances)×100 (wavelengths)×89 (reaction times).

We constructed three similarity matrices for the three modes as follows. Since 12 chemical substances differ in contents of three structural isomers of a certain chemical compound, each substance can be represented as a three-dimensional feature vector. We defined the similarity between two substances as the inverse of Euclidean distance between their feature vectors. Also, since wavelength and reaction time have continuous real values, we simply set the similarity of two consecutive wavelength values (or reaction time values) to one.

[2] The datasets are available from http://www.models.life.ku.dk/datasets

Benchmark Dataset 2: Licorice. Another benchmark dataset we use is the 'Three-way electronic nose dataset'[2], which consists of measurements of an odor sensing system applied to licorices for checking their quality, and is represented as a third-order tensor of size 18 (samples)×241 (reaction times)×12 (sensors).

Since each of 18 samples is labeled with one of the three quality labels, {'BAD','FBAD','GOOD'}, we set the similarity between two samples sharing an identical label to one. The similarity for reaction time is defined in the same way as for the flow injection dataset. Eventually, we obtained two similarity matrices for this dataset.

4.2 Experimental Settings

Comparison Methods. We compared the following 3 (regularization methods) ×2 (decomposition models) = 6 methods.

1. Ordinary {CP, Tucker}-decomposition,
2. Within-mode regularized {CP, Tucker}-decomposition,
3. Cross-mode regularized {CP, Tucker}-decomposition.

We used EM-based algorithms to impute missing values, especially, we used the one which updates missing value estimates each time a factor is updated [20], since it converged faster. We set the initial estimates for the unobserved elements of $\boldsymbol{\mathcal{X}}$ to the average of the observed elements, and those for $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ to the leading eigenvectors of $\mathbf{X}_{(1)}$, $\mathbf{X}_{(2)}$, and $\mathbf{X}_{(3)}$, respectively.

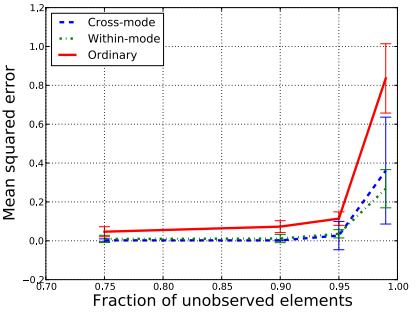
We set the model ranks as $P \equiv Q \equiv R \equiv 2$ for the synthetic dataset, $P \equiv Q \equiv R \equiv 4$ for the flow injection dataset, and $P \equiv Q \equiv R \equiv 3$ for the licorice dataset, based on the results of preliminary experiments. The hyper-parameter α was selected from $\left\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\right\}$ by using cross-validation.

Element-Wise Missing vs. Slice-Wise Missing. We test two kinds of assumptions on missing elements, that are, element-wise missing and slice-wise missing. In the element-wise missing setting, each element $[\boldsymbol{\mathcal{X}}]_{i,j,k}$ is missing independently. On the other hand, in the slice-wise missing setting, missing values occur at object level, and therefore all of the elements related to some objects are totally missing. In other words, slices such as $\{[\boldsymbol{\mathcal{X}}]_{i,j,k}\}_{j,k}$ for some i, $\{[\boldsymbol{\mathcal{X}}]_{i,j,k}\}_{i,k}$ for some j, and $\{[\boldsymbol{\mathcal{X}}_{i,j,k}]\}_{i,j}$ for some k are missing, which means that missing values occurring in a more bursty manner.

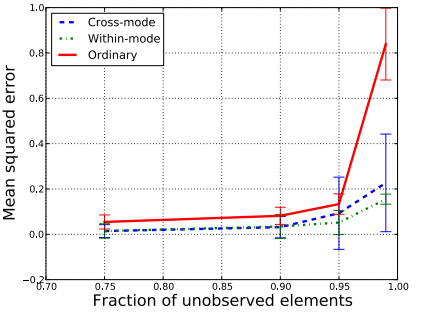
We varied the fraction of unobserved elements among $\{0.75, 0.9, 0.95, 0.99\}$ for the element-wise missing setting, and among $\{0.5, 0.75, 0.9, 0.95\}$ for the slice-wise missing setting. We randomly selected elements or objects to be used as the unobserved parts, and evaluated the mean squared errors between true values and predicted values. We continued the evaluation ten times, and recorded the averaged errors and their standard errors.

4.3 Results

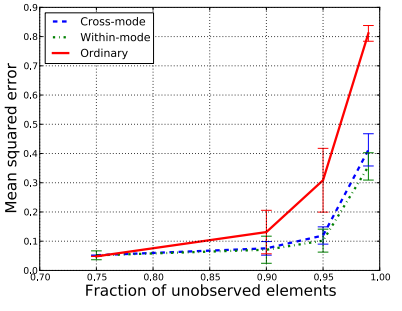
Figure 4 shows the accuracy of tensor completion by the six methods ({'Ordinary', 'Within-mode', 'Cross-mode'}×{CP-decomposition, Tucker-decomposition}) for



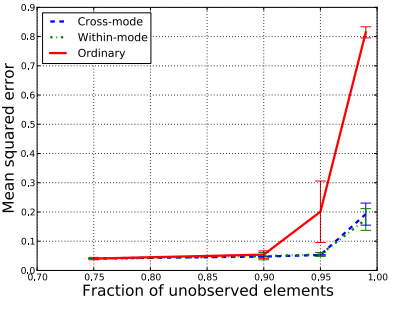
(a) CP-decompositions for the synthetic dataset



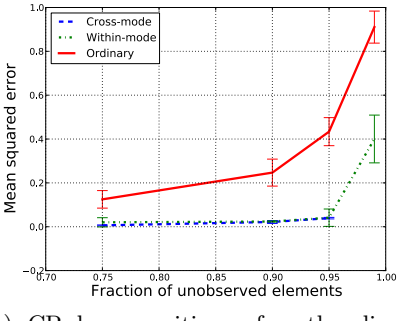
(b) Tucker decompositions for the synthetic dataset



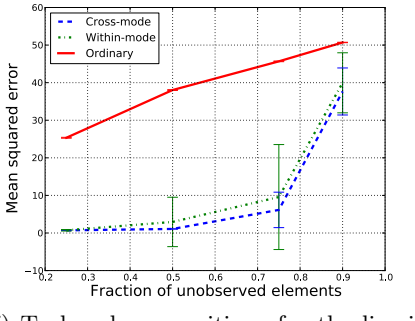
(c) CP-decompositions for the flow injection dataset



(d) Tucker decompositions for the flow injection dataset

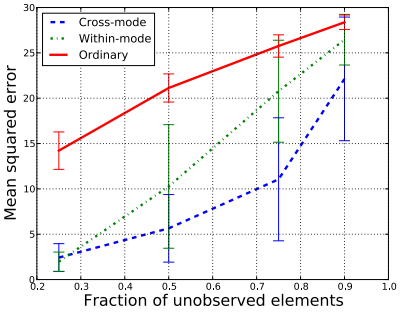


(e) CP-decompositions for the licorice dataset

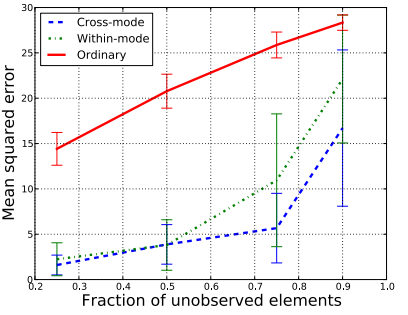


(f) Tucker decompositions for the licorice dataset

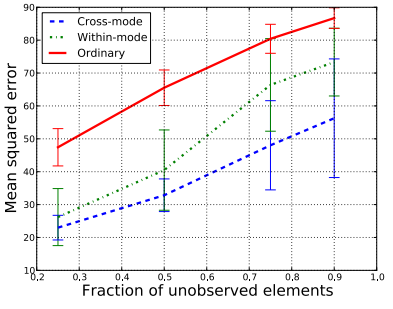
Fig. 4. Accuracy of tensor completion for three datasets in the *element-wise* missing setting. The proposed methods perform well when observations are sparse.



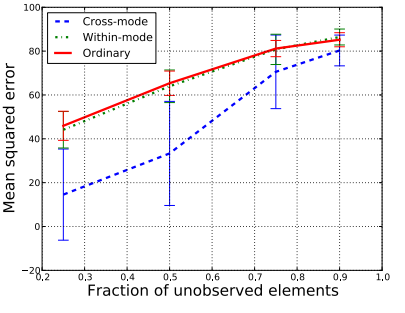
(a) CP-decompositions for the synthetic dataset



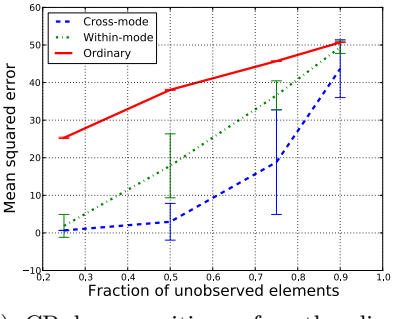
(b) Tucker decompositions for the synthetic dataset



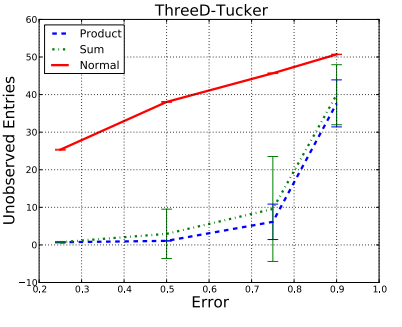
(c) CP-decompositions for the flow injection dataset



(d) Tucker decompositions for the flow injection dataset



(e) CP-decompositions for the licorice dataset



(f) Tucker decompositions for the licorice dataset

Fig. 5. Accuracy of tensor completion for three datasets in the *slice-wise* missing setting. The cross-mode regularization method performs especially well when observations are sparse.

three datasets (the synthetic dataset, the flow injection dataset, and the licorice dataset) in the element-wise missing setting. Overall, we can see that incorporating auxiliary information improves the predictive accuracy, although the ordinary tensor decomposition methods still work fairly well when the fraction of unobserved elements is less than 95%. The proposed methods perform well especially when observations are sparse.

On the other hand, Figure 5 shows the results for the slice-wise missing setting. In this case, since missing values occur at object level, reasonable inference is not possible only with the low-rank assumption, and the performance severely gets worse in sparse cases. When with auxiliary information, especially, the cross-mode regularization keeps its performance compared with the other methods even in the bursty sparse cases. This is because the cross-mode regularization uses the auxiliary information more aggressively than the within-mode regularization, that is, each element of $\mathbf{U}$ interacts with at most $I - 1$ other elements in $\mathbf{U}$ in the within-mode regularization, while it interacts with all of the other elements in all of $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ in the cross-mode regularization.

Finally, we briefly mention the computation time. Although introducing auxiliary information slightly increases the time and space complexity of the decomposition algorithms, the actual computation time was almost as same as that for ordinary decomposition methods (without auxiliary information). This is partially because we used relatively small datasets in the experiments, and further investigation with larger datasets should be made in future work.

5 Related Work

Tensor factorization methods have recently been studied extensively, and widely applied in the data mining communities. CANDECOMP/PARAFAC(CP)-decomposition [7] is a tensor factorization method that can be seen as a special case of Tucker decomposition [19] by making its core tensor super-diagonal. The CP-decomposition is applied to various problems including chemo-informatics [10]. Its variant called pair-wise interaction tensor factorization [15] accelerates its computation by using the stochastic gradient descent, and is applied to a large-scale tag recommendation problem.

There also exist probabilistic extensions of tensor factorization methods. Shashua and Hazan [17] studied the PARAFAC model under the non-negativity constraint with latent variables. Chu and Ghahramani [4] proposed a probabilistic extension of the Tucker method, known as pTucker.

Although we focus on the squared loss function (3) in this paper, changing the loss function corresponds to non-Gaussian probabilistic models of tensor factorization. In several matrix and tensor factorization studies, non-Gaussian observations have been dealt with. Collins *et al.* [5] generalized the likelihood of the probabilistic PCA to the exponential family, which was further extended to tensors by Hayashi *et al.* [8].

There are several studies to incorporate auxiliary information into matrix factorization. Li *et al.* [12] introduced a regularizer for one of factor matrices by a graph Laplacian based on geometry of data distribution. A similar approach is proposed by Cai *et al.* [3]. Lu *et al.* [13] proposed incorporated both spatial

and temporal information by using graph Laplacian and Kalman filter. Adams *et al.* [2] extended the probabilistic matrix factorization [16] to incorporate side information. In their work, Gaussian process priors are introduced to enforce smoothness to factors. Some work use auxiliary information not in regularization terms but as bias variables added to model parameters [21,14]. To best of our knowledge, our work is the first attempt to incorporate auxiliary information into tensor factorization.

6 Conclusion and Future Work

In this paper, we proposed to use relationships among data as auxiliary information in addition to the low-rank assumption to improve accuracy of tensor factorization. We introduced two regularization approaches using graph Laplacians induced from the relationships, and designed approximate solutions for the optimization problems. Numerical experiments using synthetic and real datasets showed that the use of auxiliary information improved completion accuracy over the existing methods based only on the low-rank assumption, especially when observations were sparse.

Although the focus of this paper is to show the usefulness of auxiliary information for tensor factorization, we will address its computational aspects extensively in future. Indeed, scalability is an important issue. In real data such as EEG data, tensors can be huge and of high dimensionality, and we need fast and memory efficient algorithms. For example, Acar *et al.* [1] eliminate some of the observed elements of large data tensors. They report entire information of the data tensor is not necessary when the tensor is of low-rank, and only a small fraction of elements ($\sim 0.5\%$ for a $1,000 \times 1,000 \times 1,000$ tensor) are sufficient for reconstruction. The idea might also work well in our approach, and the regularization using auxiliary information might further reduce the sufficient number of elements. Memory efficiency is also an important factor, since the number of parameters are linearly dependent on the dimensionality of each mode. Kolda and Sun [11] use the sparsity of tensor, and dramatically reduce the memory space as 1,000 times smaller than the original algorithm for Tucker decomposition. The stochastic gradient optimization approach [15] would be also promising.

References

1. Acar, E., Dunlavy, D.M., Kolda, T.G., Mørup, M.: Scalable tensor factorizations with missing data. In: Proceedings of the 2010 SIAM International Conference on Data Mining, pp. 701–712 (2010)
2. Adams, R.P., Dahl, G.E., Murray, I.: Incorporating side information into probabilistic matrix factorization using Gaussian processes. In: Grünwald, P., Spirtes, P. (eds.) Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, pp. 1–9 (2010)
3. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized non-negative matrix factorization for data representation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2010)

4. Chu, W., Ghahramani, Z.: Probabilistic models for incomplete multi-dimensional arrays. In: Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (2009)
5. Collins, M., Dasgupta, S., Schapire, R.E.: A generalization of principal components analysis to the exponential family. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge (2002)
6. Dunlavy, D.M., Kolda, T.G., Acar, E.: Temporal link prediction using matrix and tensor factorizations. ACM Transactions on Knowledge Discovery from Data 5, 10:1–10:27 (2011)
7. Harshman, R.A.: Foundations of the PARAFAC procedure: models and conditions for an "explanatory" multi-modal factor analysis. UCLA Working Papers in Phonetics 16(1), 84 (1970)
8. Hayashi, K., Takenouchi, T., Shibata, T., Kamiya, Y., Kato, D., Kunieda, K., Yamada, K., Ikeda, K.: Exponential family tensor factorization for missing-values prediction and anomaly detection. In: Proceedings of the 10th IEEE International Conference on Data Mining, pp. 216–225 (2010)
9. Kashima, H., Kato, T., Yamanishi, Y., Sugiyama, M., Tsuda, K.: Link propagation: A fast semi-supervised algorithm for link prediction. In: Proceedings of the 2009 SIAM International Conference on Data Mining (2009)
10. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Review 51(3), 455–500 (2009)
11. Kolda, T.G., Sun, J.: Scalable tensor decompositions for multi-aspect data mining. In: Proceedings of the 8th IEEE International Conference on Data Mining, pp. 363–372 (2008)
12. Li, W.-J., Yeung, D.-Y.: Relation regularized matrix factorization. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp. 1126–1131 (2009)
13. Lu, Z., Agarwal, D., Dhillon, I.S.: A spatio-temporal approach to collaborative filtering. In: Proceedings of the 3rd ACM Conference on Recommender Systems, pp. 13–20 (2009)
14. Porteous, I., Asuncion, A., Welling, M.: Bayesian matrix factorization with side information and Dirichlet process mixtures. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence, pp. 563–568 (2010)
15. Rendle, S., Thieme, L.S.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining, pp. 81–90 (2010)
16. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems, vol. 20. MIT Press, Cambridge (2008)
17. Shashua, A., Hazan, T.: Non-negative tensor factorization with applications to statistics and computer vision. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 792–799 (2005)
18. Srebro, N.: Learning with matrix factorizations. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA (2004)
19. Tucker, L.: Some mathematical notes on three-mode factor analysis. Psychometrika 31(3), 279–311 (1966)
20. Walczak, B.: Dealing with missing data Part I. Chemometrics and Intelligent Laboratory Systems 58(1), 15–27 (2001)
21. Yu, K., Lafferty, J., Zhu, S., Gong, Y.: Large-scale collaborative prediction using a nonparametric random effects model. In: Proceedings of the 26th International Conference on Machine Learning, pp. 1185–1192 (2009)

Kernels for Link Prediction with Latent Feature Models

Canh Hao Nguyen and Hiroshi Mamitsuka

Bioinformatics Center, ICR, Kyoto University,
Gokasho, Uji, Kyoto, 611-0011, Japan
{canhhao,mami}@kuicr.kyoto-u.ac.jp

Abstract. Predicting new links in a network is a problem of interest in many application domains. Most of the prediction methods utilize information on the network's entities such as nodes to build a model of links. Network structures are usually not used except for the networks with similarity or relatedness semantics. In this work, we use network structures for link prediction with a more general network type with latent feature models. The problem is the difficulty to train these models directly for large data. We propose a method to solve this problem using kernels and cast the link prediction problem into a binary classification problem. The key idea is not to infer latent features explicitly, but to represent these features implicitly in the kernels, making the method scalable to large networks. In contrast to the other methods for latent feature models, our method inherits all the advantages of kernel framework: optimality, efficiency and nonlinearity. We apply our method to real data of protein-protein interactions to show the merits of our method.

1 Introduction

Link prediction is a major problem in relational data learning [6]. In Social Networks and Collaborative Filtering, one needs to suggest links for entities for recommendation. In Bioinformatics and Chemoinformatics, potentially valid links such as interactions are required to speed up experimental processes. In order to predict links in relational data, one needs to provide a common model for both entities and relationships (such as links) in the data. As these two types of information are of different natures, models are difficult to design and learn. While modeling entities are of common practice, modeling relationships is usually of more difficulty. For link prediction, these relations are interpreted differently and reflect different semantics. While in social networks, links are usually of similarity or relatedness nature, it is not the case elsewhere. It is the target of this work to deal with a more general type of network structures, to build link models and to train them on large-sized real data.

In principle, there are two different kinds of information used to learn a link model for link prediction. One is the information of network entities such as nodes of the networks. The methods falling in this category usually use the information of a pair of nodes to induce the label of having or not having a link

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 517–532, 2011.

[2,8,3]. Typical examples are sequences or profile information of genes, which are used to predict links (edges) in their networks. By using only the information on networks' entities, the models of the networks assume an independent and identical distribution of links. In other words, the structures of the links themselves are completely ignored. This is an unrealistic assumption in many domains where networks' structures themselves have patterns, such as social networks [12,13] and biological networks [18,1]. It is an objective of this work to show that network structures themselves contain information for the task.

The second kind of information used to predict links is structures (topologies) of the networks themselves. Similarity networks, due to its analogy to kernels [10,7,17,16,11], can be modeled with kernels and therefore, there exist scalable methods. Bipartite networks can also rely on kernels for each of their parts together with aligning the parts [19]. However, it is more difficult to deal with networks without similarity or relatedness semantics. A common network type is modeled with latent feature models [14,1]. This network model includes similarity networks and bipartite networks as special cases. For this type of networks, the available models are usually the plain latent feature models or matrix based models [5,1,13]. Training these models usually requires to generate latent features explicitly. This is a very time-consuming process, usually does not applicable to medium-sized networks. Another problem is that approximation in the training process leads to suboptimal solutions. Without approximation, these models do not scale to the sizes of real data, even for medium-sized networks. It is our motivation to be able to learn a model of this general type of networks efficiently and to avoid suboptimal solutions.

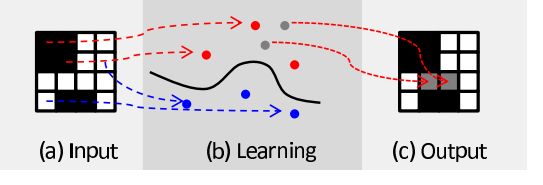


Fig. 1. Given an adjacency matrix as input, the method embeds cells (links or nonlinks) into a feature space and using classification to infer new links

We provide a kernel method to link prediction problem using network structures for the network structures modeled with latent features. The overall description of the method is depicted in Figure 1. We design kernels to encode the structures implicitly, without the need of generating the latent features themselves. The idea is to give high kernel values to the pairs of links that potentially share latent features. We show how suitable the kernel is to sparse networks. By not inducing the latent feature directly, our method is much faster compared to the long execution times faced by the methods that explicitly induce latent features. Our method also gives a globally optimal solution as opposed to the other methods. Nonlinearity can be incorporated into the model seamlessly. For

these advantages, our method gives a very high predictive performance for link prediction problems and applicable to real datasets of large sizes.

The paper is organized as follows. We describe the generative model of networks' links with latent features in Section 2. We then develop our kernels for this model in Section 3. We visualize the idea of the kernels for this model in Section 4. We show our application in protein-protein interaction networks in Section 5 and conclude the paper.

2 Latent Feature Models of Graphs

2.1 Biological Motivation

An example is that a protein (node) is a collection of domains (features). A protein-protein interaction (PPI) is caused by an interaction between two domains from the two proteins [4]. However, the knowledge of domain may be incomplete, and domain-domain interaction is far less understood. Therefore, we wish to incorporate the domain-domain interaction knowledge in an implicit way. Given enough links, we want to infer from many pairs of interacting proteins' common features that play the role of domains and some pairs of features are likely to interact for PPI task.

2.2 Latent Feature Models of Graphs

We describe a latent feature model of graphs as also appeared in [13]. In this model, a link (edge) in the graph is generated by the relation between the latent features of the adjacent nodes. Denote $A \in \mathbb{R}^{n \times n}$ as the adjacency matrix of the graph. In general, A can be any real matrix. For our special purpose of modeling undirected networks, we assume that A is a binary symmetric matrix. Denote $F \in \mathbb{R}^{n \times d}$ as a binary matrix where each row is a d-dimensional vector of latent features. Denote $W \in \mathbb{R}^{d \times d}$ as a real matrix encoding strength of feature interactions. That is, W_{ij} encodes the strength of interaction between the i^{th} feature and the j^{th} feature. We define that (F,W) is a latent feature model for A when

$$A = FWF^T. \tag{1}$$

It is possible that there are several causes for a link but we only record one link (existence of a link rather than multiplicity of the link). Therefore, the equality in the equation (1) is replaced by the element-wise operator $\min(x, 1)$ where x is the entry of the right hand side of the equation (1). It can be rewritten as follows

$$A = \min(FWF^T, 1). \tag{2}$$

Simulation Example: The idea of latent feature model is depicted in Figure 2. In this example, the set of nodes have three latent features, with the first two nodes has two features. As for the feature interaction matrix, it indicates that the first and the second features interact, also the third feature interact with itself.

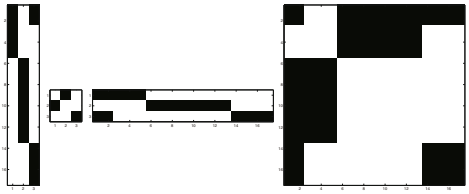


Fig. 2. $FWF^T \rightarrow A$: the adjacency of the graph (rightmost) is generated by the three matrices: the latent feature matrix (F - leftmost), the feature interacting matrix (W) and the transpose of the latent feature matrix (F^T). A black cell indicates a positive entry and a white cell indicates a 0 entry. In our latent feature model, product of the first three matrices generates the last one.

For the benefits of using latent feature models instead of latent class models, we refer to [13] for details. The following properties are observed.

- Each entry of W generates a subgraph given F. The overall generated graph is a superimposition of these subgraphs. This makes the model an additive one.
- If W is diagonal, then the generated graph can be decomposed as a set of cliques. This is a model of similarity graph.
- If W is symmetric, then the generated graph is symmetric. It is usually used to encode undirected graphs.
- If F has exactly one nonzero entry in each row (such as the ones generated by Chinese Restaurant Processes (CRP)), then generating F is equivalent to clustering of nodes.
- If the nodes can be divided into two groups, each with a separate set of features, and W has only interactions of features from the two different sets, then the generated graph is bipartite.

These properties make latent feature models general generative models for graphs.

2.3 Ideal Kernels

Given a latent feature model as a generative model of graphs, one wish to define a kernel that encodes the similarity of the nodes using this model. The semantic similarity of two nodes under this model is the amount of latent features they share, weighted by the importance of each feature. Therefore, we define the *ideal kernels* as follows.

Definition (Ideal kernels). Define the set of ideal kernels for a latent feature model of nodes F to be $\mathbb{K} = \{K^*(D)\}_D$, with $D \in \mathbb{R}^{d \times d}$ is any diagonal matrix with positive entries, and

$$K^*(D) = FDF^T. \tag{3}$$

A kernel value $K^*(D)_{ij} = F_i D (F_j)^T$ is the weighted sum of the number of common feature between the i-th and j-th nodes. However, since latent feature

models are hardly obtained, the ideal kernels are not available. Any kernel defined from data with latent feature models should be close to some ideal kernels in some senses.

3 Link Kernels with Latent Features

We describe our kernel method to link prediction given the latent feature model assumption of the graph structures. The idea is to use the model to derive a kernel between all pairs of the nodes (called link kernel). We define the following terms: a link is considered as a *positive* pair of nodes while a *negative* pair of nodes encodes the non-existence of a link (nonlink).

Input: *Adjacency matrix A.*

1. *Construct a node kernel K_n following latent feature models.*
2. *Construct a link kernel K based on K_n.*
3. *Learn a SVM on K.*

Output: *New adjacency matrix based on the SVM.*

Fig. 3. Network Structure based Link Prediction

The link prediction problem is then formulated as a binary classification problem. In the end, we classify the two classes, link class and nonlink class using Support Vector Machines. The overall strategy is in Figure 3.

The kernel between pairs is based on the kernel between nodes themselves (called node kernel). We first describe the node kernel (K_n) that encodes the latent feature model, its relations to ideal kernels in sparse graphs, and then the link kernel.

3.1 Node Kernels with Latent Features

As latent feature models are the generative models of graphs, we wish to define the similarity of two nodes as the likelihood of having common latent features. This is inherently different from similarity models where similarity means the likelihood of reaching the other node through random walks [12]. However, latent features are implicit, we must estimate the similarity, in form of kernels, between nodes empirically. Knowing that nodes with common features tend to link to common neighbors in latent feature models, we define the basic node kernels as the amount of common neighbors of the nodes, as follows.

$$K_n = norm(AA^T), \tag{4}$$

where *norm* indicates the normalization operator to make the diagonal elements all 1. Specifically,

$$(K_n)_{ij} = \frac{(AA^T)_{ij}}{\sqrt{(AA^T)_{ii}(AA^T)_{jj}}}. \tag{5}$$

A special case of this kernel is when all the nodes have exactly one feature (such as generated by CRP), then $K_n(a, b)$, for any node a and b, is either 1 or 0, indicating if they have the feature in common or not. An implication is that in networks where latent features are expected to be sparse, K_n behaves similarly to this extreme case, showing a good indication of common latent features.

Given the basic node kernel, additional tricks can be applied on top of this kernel to make new families of kernels. Diffusion kernels and exponential kernels on top of K_n still conserve the idea of latent features. Of course, the higher the exponential we take on K_n the looser the idea can be kept. However, please note that the most general version of spectral transform [11] is not guaranteed to conserve the latent feature assumption.

Note that this node kernel can be one of many terms in other kernels for similarity graphs such as path-based [15,10,7,12]. However, we find that our proposed kernel in particular encodes the latent feature model that is suitable for our problem.

3.2 Relation to Ideal Kernels on Sparse Graphs

When the graphs are sparse (in our applications), sparse models are required to model them. The following propositions show the relationship between the node kernel K_n and the ideal kernels when the models are sparse. Denote S_i as the set of latent features of node i.

Proposition 1 ***(Positivity).*** *If $S_i \cap S_j \neq \emptyset$ (two nodes i and j share at least one latent feature) and that feature interacts with another feature, then $(AA^T)_{ij} > 0$, therefore $(K_n)_{ij} > 0$.*

Proof. The proof can be easily seen because when they share a feature, they have a common nonempty neighborhood and therefore, $(K_n)_{ij} > 0$. □

This shows that if the values of ideal kernels on two nodes are positive, the value of K_n is positive as well. This means that whenever the two nodes are similar (positive kernel value) in the model, the kernel K_n can recognize that. This is useful for sparse graphs (causing sparse kernels), K_n is not sparser than any ideal kernel.

Proposition 2 ***(Monotonicity).*** *Suppose that the edges have the same amount of neighbors in the sense that $(AA^T)_{ii}(AA^T)_{jj} = (AA^T)_{kk}(AA^T)_{ll}$. If $(S_i \cap S_j) \supseteq (S_k \cap S_l)$ then $(K_n)_{ij} \geq (K_n)_{kl}$.*

The first assumption is about the equal amounts of neighbors for the two pairs of nodes. The amount of neighbors of a pair of nodes is defined to be the product

of numbers of its adjacent nodes. The conclusion is that, the more latent features they share, the higher the kernel value is. This is a property of the ideal kernels by the way they are defined, showing an analogy of K_n to ideal kernels.

Proof. As $(S_i \cap S_j) \supseteq (S_k \cap S_l)$, the common neighborhood of the nodes i and j is a superset of the the common neighborhood of the nodes k and l. Therefore, $(AA^T)_{ij} \geq (AA^T)_{kl}$.

$$(K_n)_{ij} = \frac{(AA^T)_{ij}}{\sqrt{(AA^T)_{ii}(AA^T)_{jj}}} \geq \frac{(AA^T)_{kl}}{\sqrt{(AA^T)_{ii}(AA^T)_{jj}}} = (K_n)_{kl} \tag{6}$$

from the definition of the node kernel in (4). □

Lemma 1 ***(One feature interaction).*** *For any latent feature model (F,W), there exists another latent feature model ($F^{'}$,$W^{'}$) that gives (i) the same adjacency matrix, (ii) the same set of ideal kernels and (iii) the feature interaction matrix is nonzero on at most one of its entries in any row and column.*

In other words, there exists another mathematically equivalent model giving the same ideal kernel sets and adjacency matrix in which each feature interacts with only one another feature.

Proof. The idea of the proof is to place a nonzero w_{ij} in one new row and column of $W^{'}$ and duplicate the columns of F when necessary to make $F^{'}$, keeping the adjacency matrix unchanged. Supposed that the (unnormalized) adjacency matrix A is computed by

$$A = FWF^T = \sum_{ij} w_{ij} F_{\cdot i}(F_{\cdot j})^T, \tag{7}$$

where $F_{\cdot i}$ is the i-th column of F.

Denote $I = \{(i,j)\}$ that $w_{ij} \neq 0$ then

$$A = \sum_{(i,j)\in I} w_{ij} F_{\cdot i}(F_{\cdot j})^T. \tag{8}$$

We then construct the $F^{'}$ and $W^{'}$ by sequentially appending feature columns and feature interaction matrix values in the formula 8 as follows.

1. Mark all the indices in I as available.
2. Initialize $F^{'}$ and $W^{'}$ to empty matrices.
3. Go the the next available index in I, pick the pair w_{ij}, then
   - If $i = j$, meaning that w_{ii} indicates a self interacting feature, then append the feature vector $F_{\cdot i}$ at the end of the already constructed $F^{'}$. Append a new row and column of $W^{'}$ with the only nonzero element w_{ij} on the diagon of $W^{'}$. Mark the index of w_{ij} in I unavailable. Repeat the process from step 3.

- If W is symmetric (A is symmetric) then $w_{ij} = w_{ji}$. Append the feature vectors $F_{.i}$ and $F_{.j}$ to the end of $F^{'}$. Suppose the size of $W^{'}$ is k, then append two new rows and columns of $W^{'}$ with the only nonzero elements are w_{ij} at $W^{'}_{k+1,k}$ and $W^{'}_{k,k+1}$. Mark the indices of w_{ij} and w_{ji} in I unavailable. Repeat the process from step 3.
- If W is not symmetric then append the features as the symmetric case to $F^{'}$. For $W^{'}$, only one nonzero element is added to $W^{'}_{k,k+1}$. Mark the index of w_{ij} in I unavailable. Repeat the process from step 3.

By doing this, then

$$F^{'}W^{'}(F^{'})^T = \sum_{(i,j)\in I} w_{ij}F_{.i}(F_{.j})^T = FWF^T = A. \tag{9}$$

Also, each row or column of $W^{'}$ has at most one nonzero entry by the way $W^{'}$ is constructed. Since the set of columns of $F^{'}$ is the same as F's, the set of ideal kernels are the same (only different by feature weighting). □

Hence, we have constructed a new model with a feature either interacts or is interacted with at most one another feature. This is to say that there is a mathematically equivalent model that each feature only interacts with one another feature. We use this fact to facilitate our sparsity reasoning as follows.

Proposition 3 ***(Ideal condition).*** *K_n is an ideal kernel ($K_n \in \mathbb{K}$) if WF^T has all row vectors uncorrelated.*

Proof. When WF^T has row vectors uncorrelated, then: $WF^T \times (WF^T)^T$ is diagonal. Denote $D = WF^T \times FW^T$, then:

$$K_n(A) = FWF^T \times FW^TF^T = FDF^T. \tag{10}$$

Since D is diagonal with nonnegative entries, $K_n \in \mathbb{K}$. □

Corollary 1. *If W is an unmixing matrix of an Independent Component Analysis model for F^T, then $K_n \in \mathbb{K}$.*

Corollary 2. *If each node has only one latent feature, such as F is generated by a Chinese Restaurant Process or any class-based model in a model that W has only one nonzero entry in each row or column (guaranteed by Lemma 1), then $K_n \in \mathbb{K}$.*

This is from the fact that WF^T has only one nonzero entry in each column, therefore any pair of row vectors would have no nonzero entries in common. This makes row vectors of WF^T uncorrelated.

When the model is sparse, K_n is close to an ideal kernel as follows. Denote $E = F^TF$, therefore $E_{lk} = (F_{.l})^TF_{.k}$ is number of nodes having both features l and k. When $l \neq k$, we expect E_{lk} to be small for sparse models. Given the Lemma 1, we assume that W_{il} and W_{jk} are the only nonzero entries in rows i

and j. Recall that $K_n = FDF^T = FWEW^TF^T$. Since the entries of W are scalars, $D_{ij} = W_{il}W_{jk}E_{lk}$ means that $D_{ij}^2/(D_{ii}D_{jj}) = E_{lk}^2/(E_{ll}E_{kk})$. When F is sparse, off-diagonal elements of E is much smaller than diagonal ones (*diagonally dominant*), the same thing can be said for D. This means that D is as diagonally dominant as E. We show quantitatively how close D is to an ideal kernel.

Proposition 4 ***(Approximation).*** *When the model is sparse in the sense that* $(\sum_{i\neq j} |W_{il}W_{jk}E_{lk}|^p)^{\frac{1}{p}} \leq \delta$ *for some small* $\delta \in \mathbb{R}$, $p \geq 0$, *there exists an ideal kernel* $F\hat{D}F^T$ *that is close to* K_n *in the sense that* $\| D - \hat{D} \|_p \leq \delta$.

In the condition $(\sum_{i\neq j} |W_{il}W_{jk}E_{lk}|^p)^{\frac{1}{p}}$, an entry $W_{il}W_{jk}E_{lk}$ is the weighted the number of nodes having the two features l and k (should also be small for sparse models, $l \neq k$ inplies $i \neq j$). $\| \cdot \|_p$ is the p-norm of a matrix.

Proof. We construct $\hat{D}$ as a diagonal matrix with $\hat{D}_{ii} = D_{ii} = W_{il}^2 E_{ll}$. Given that $D_{ij} = W_{il}W_{jk}E_{lk}$, then

$$\| D - \hat{D} \|_p = (\sum_{i\neq j} |D_{ij}|^p)^{\frac{1}{p}} = (\sum_{i\neq j} |W_{il}W_{jk}E_{lk}|^p)^{\frac{1}{p}} \leq \delta. \tag{11}$$

This shows that the more sparse the model, the closer D is to ideal kernels. □

When a model is *sparse* in the sense that each node has very few latent features, each latent feature interacts with one another features (by the Lemma 1), K_n should be close to the ideal kernel $F\hat{D}F^T$ since $F\hat{D}F^T$ is linear in $\hat{D}$.

All these properties make the kernel K_n a good approximation of ideal kernels in sparse models. Sparse models are suitable for our applications in sparse networks (nodes with small degrees). We elaborate more in the application part.

3.3 Link Kernels with Latent Features

Given the node kernel K_n, a kernel between two links is usually defined to be the combined similarity between the pairs of nodes of the links. We choose the widely used and experimentally justified tensor product pairwise kernel [2] to be the kernel for pairs as follows.

$$K((a,b),(c,d)) = K_n(a,c) \cdot K_n(b,d) + K_n(a,d) \cdot K_n(b,c). \tag{12}$$

Here is the kernel for the two pairs of nodes (a,b) and (c,d). Denote a_i, b_j as the features of a and b respectively in the feature space of K_n, then the feature space of K consists of the following features for a pair (a,b):

$$a_i b_j + a_j b_i, \tag{13}$$

as in [2]. Given that the node kernel K_n is supposed to be the likelihood of having common latent features, link kernel indicates the chance of having two pairs of nodes with common features. For example, when the pair $\{a,c\}$ share common features and so do the pair $\{b,d\}$, $K((a,b),(c,d))$ is high.

Nonlinearity can be incorporated into this kernel, such as Gaussian kernels on top of it.

4 Demonstration

We show the idea of our kernel using the simulation example in the previous section. Suppose that we observe an incomplete graph of the example as in the adjacency matrix in Figure 4 (80% of the links are observed). We show step by step the idea of node kernels and latent features.

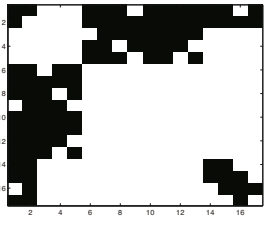


Fig. 4. Adjacency matrix of an incomplete graph from the simulation example

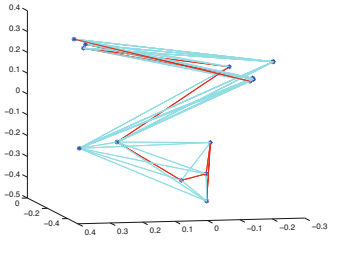


Fig. 5. Visualing the graph with the node kernel. The cyan edges are the observed links of the graph while the red edges are the missing ones according to the model. We observed that the red ones follow the same patterns as the cyan ones.

Node Kernel: We visualize the nodes using the node kernel defined in (4). We use kernel PCA and plot the nodes using the first three components in Figure 5. The nodes in the figure are the nodes of the graphs. The cyan edges are the observed links. The red edges are the missing links according to the model. First, we can observe that the cyan edges follow certain patterns of direction and endpoints. Another point can be observed is that the red edges follows the same patterns (endpoints lie in clusters, edges connect the same cluster pairs). This is exactly what we want in learning, testing data having the same distribution as training data. What left to be done is to keep these patterns in some spatial representations of the edges.

Latent Features: We look into the model to show the distribution of nodes with features. We label the nodes with the same feature in shaded ellipses in Figure 6. The labels of the eclipses, $F1$, $F2$ and $F3$ correspond to the three latent features in the example. We can observe that the node kernel makes these nodes close to each other. The nodes with two features ($F1$ and $F3$) lie somewhere in between the nodes with only one of the two features ($F1$ or $F3$).

Negative Links: As shown in Figure 5 that the observed edges and missing edges have similar patterns. However, we use SVM to classify, we also wish the negative class (nonlink) not to follow the patterns. Therefore, we show all the edges that belong to the nonlink class in Figure 7. We can observe that they do not follow that patterns of the positive class (the same endpoints but connecting different cluster pairs from positive links').

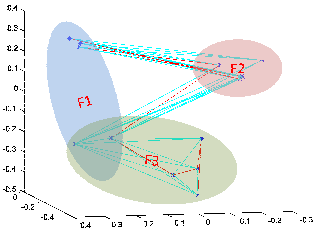


Fig. 6. The positive class of links. The nodes are grouped by their common latent features. It shows that nodes sharing more common features tend to group together.

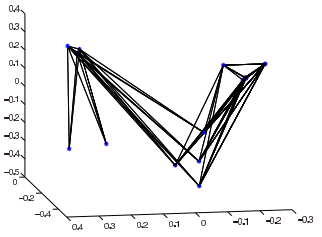


Fig. 7. The set of nonlink class. Compared to the link class, the nonlink class has totally different positions and orientations.

The demonstration shows that our designed node kernel successfully discovers the patterns of the link class as opposed to the nonlink class. It tends to group the nodes with common features close to each other as we designed. We wish to clarify the difference of our method from the other methods such as using Chinese Restaurant Processes or Indian Buffet Processes [14,9,13]. These methods group these nodes together explicitly while our method use a *soft* version of putting them close in a space. We believe that this is the key to be robust, allowing the inference step to be globally optimal and computationally efficient.

5 Application on Non-similarity Networks with Latent Features

Our targeted application is to model network structures with the latent feature models. Even though the model includes similarity networks as special cases, our target is to model more difficult ones, which are not similarity ones. Social networks are usually similarity ones, therefore not the prime target of our method. The PPI networks are the typical examples as it is non-similarity (we also show the experiments for this in the following subsection). There are other networks studied in Bioinformatics but they are too sparse and not large enough to study structures statistically.

We used the PPI networks of yeast and fruit-fly from DIP database for their largest hand-curated networks available. For statistical study, we attracted only the largest connected component of the network in which the degree of each node is not less than m (minimum degree of nodes). In our experiments, we show all the values of m from 1 to 10 (8 for fruit-fly). For yeast, the subnetwork with $m = 1$ has 4762 nodes and 21836 links, and the subnetwork with $m = 10$ has 756 nodes and 8924 links. For fruit-fly, the subnetwork with $m = 1$ has 6644 nodes and 21501 links, and the subnetwork with $m = 8$ has 713 nodes and 5243 links. This is the size of data that we expect and we will show latter on that the other methods based on latent feature inference do not scale to this size.

Real PPI networks are sparse: less than 20% of the nodes in yeast and none in fruit-fly have degrees of 10 or more. This is the reason for sparsity analysis in Section 3.2. The sparse networks require the models to be sparse as well.

For each subnetwork, we used SVM ($C = 0.001$, but the results are the same for a range of C from 0.0001 to 0.1) as the classifier for our link kernels. We showed the average AUC score of different train/test splits with the ratio 90/10. We first showed the appropriateness of the assumption of latent features as opposed to the usual assumption of similarity. We then showed the time required to build one model to reflect the difference in execution times or our method compared to Indian Buffet Process (IBP) as described in [13] (with parameter $\alpha = 3$, which we found to be a good trade-off to be able to train on our smallest datasets and induce a model with enough number of latent features). We then showed the performance of link kernels in link prediction. We also compared to sequence based link prediction to show the advantage using network structures.

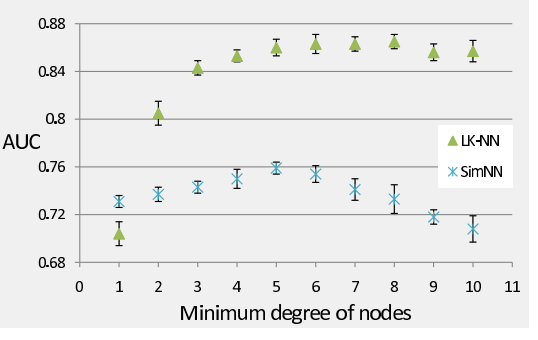


Fig. 8. Latent feature versus similarity assumption: AUC of the direct method for link prediction (vertical) at different minimum degree of the network (horizontal axis)

5.1 Latent Feature versus Similarity

We show the fitness of the latent feature model on yeast PPI networks as opposed to the similarity model in Figure 8. We used the direct method [18] to predict links based on the two models. The idea of the direct method is simply nearest neighbor classifiers. For similarity assumption, probability of having link between two nodes is proportional to how similar they are in their connectivity patterns (called SimNN). On the other hand, with latent feature model, probability of a link between two nodes is determined by the class of its nearest neighbor in the *link kernel* (called LK-NN), not the kernel between nodes.

The figure showed a significantly different AUC scores of the two methods on yeast's networks. LK-NN was usually much higher (around 0.1 and more). It showed that latent feature model was more suitable to PPI networks than the conventional similarity one. The exception was at the subnetwork with the minimum degree of nodes of one. Since we were using only network structures, the nodes with degree one may not contribute to the network structure in these

models. Therefore, results for the subnetworks with larger nodes' degrees demonstrated the reasonability of the assumptions used. For this reason, the latent feature assumption was more reasonable than similarity assumption here, and the methods for similarity networks were not recommended to use.

5.2 Execution Time

We show execution time required to build one model using our link kernels in Figure 9. To compare to IBP, we also show the execution time in the process of building one model before it *burns in* in Figure 10. For the long times required by IBP, we only show the execution time for the smallest subnetwork (minimum degree of 10 with only 756 nodes), which is supposed to require the least time among all the subnetworks.

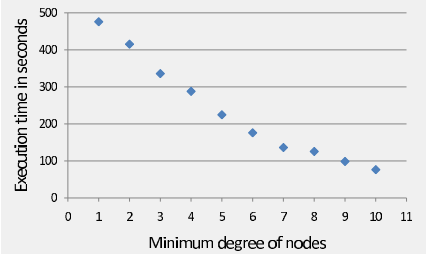


Fig. 9. The time required to build one model using link kernels (in seconds) at subnetworks with different minimum degrees. Note that for the smallest subnetwork, the execution time is less than 90 seconds.

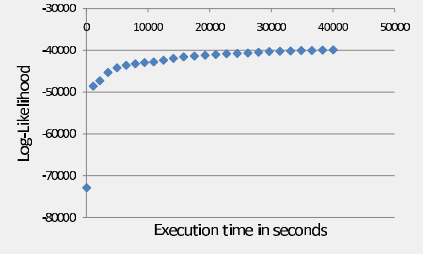


Fig. 10. The time to train an IBP model of the smallest subnetwork with $m = 10$. The vertical axis is the log-likelihood obtained from the model during the training process as a function of time in the horizontal axis.

We can see that the link kernels required less than 90 seconds for the smallest subnetwork, and increased to less than 500 seconds for the largest subnetwork of 4762 nodes. On the other hand, IBP required many hours for the smallest subnetwork of 756 nodes and did not burn-in on larger ones within one day. We conclude that our method using kernel saved many orders of magnitude the time to train one model, making training on medium-sized networks possible. This is a key advantage of our kernel method.

5.3 Link Prediction Results

We compared the prediction ability of our method using link kernel to the baseline of using IBP as in [13] in Figure 11. The results are for different subnetworks with different minimum degrees. For small minimum degrees, the subnetworks have higher coverage on the whole network while the large minimum degrees

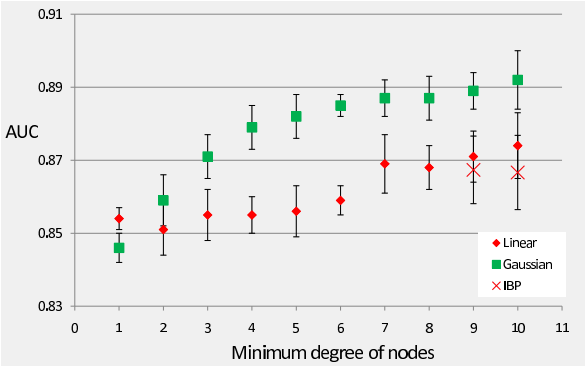


Fig. 11. Link prediction results on yeast PPI network: AUC of link prediction with different methods (vertical) at different minimum degrees of the network (horizontal axis). Note that IBP does not scale with larger datasets, its results are not available.

will extract denser parts of the network, making statistical inference on this part more reliable. We show two versions of our link kernels: linear kernels and Gaussian kernels ($\gamma = 2$). The incomplete results of IBP was due to the fact that experiments took too much time (more than one day) to train one model.

We can read from yeast's results in Figure 11 that linear kernels had similar AUC scores with IBP. However, when using the nonlinear version of Gaussian kernels, AUC scores were significantly higher. We conclude that our kernel based method provided a significantly higher AUC scores than that of IBP. One surprise was that even using only network topology, we archived high AUC scores close to 0.9. These scores were much higher than random prediction. Given that the PPI networks are known to be noisy and incomplete, this experiment showed that there are patterns of the topology of the PPI networks. It also showed that our method was effective to encode these patterns.

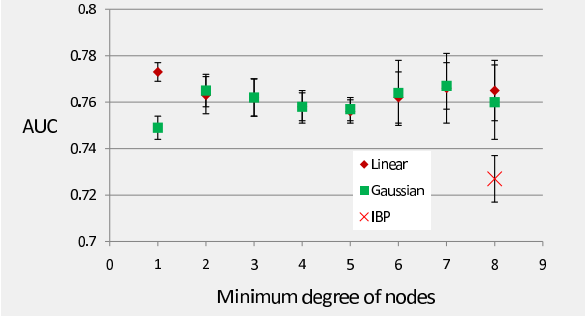


Fig. 12. Link prediction results on fruit-fly PPI network: AUC of link prediction with different methods (vertical) at different minimum degrees of the network (horizontal axis). IBP results are missing due to their time consumption.

Similarly, we can see in the fruit-fly's results in Figure 12 that our method based on kernels outperformed IBP. The results on fruit-fly's networks were much lower than on yeast due to the fact that fruit-fly's ones are sparser, involving more proteins and there are no proteins with degrees of 10 or more.

5.4 Comparison to Sequence-Based Prediction

As opposed to using network structures, traditional methods use nodes' information such as protein sequences. Therefore, we also compared to spectrum kernels on sequences to predict links in the same manner. The results were not shown in the Figure 11 and 12 because they ranged differently. The highest AUC score on any subset for yeast was 0.71 ± 0.008 and 0.65 ± 0.016 for fruit-fly. We observe that network structures gave statistically significantly higher AUC scores (with t-tests at 0.01 level). This might be due to the fact that kernels based on sequences contain too much redundant information, since sequence-based kernels use all k-mers across the protein sequences. Sequence based kernels are redundant as only small parts determine its interaction ability to others.

6 Conclusion

We studied the problem of predicting new links using network structures in a more general type of networks. Specifically, we studied the networks that can be modeled by generative processes with latent features. This is a more general model of networks than the usually assumed similarity networks in most of the applications. In order to model real networks of medium or large size, we used kernels and casted the problem as a supervised learning one, inheriting optimality, efficiency and nonlinearity of the kernel framework. We showed the suitability of the kernels on sparse networks. We applied to the non-similarity networks of protein-protein interactions. The results showed that our kernel-based method gave a much higher performance than direct latent feature inference by IBP. Our method was also many orders of magnitude faster, and scaled to the sizes of real networks, unlike IBP. It was also shown that network structures gave higher results than information of the nodes. We conclude that for sparse networks with latent feature models, our method is able to utilize the relevant information in network structures to give faster, more scalable solutions, and significantly higher performances.

Acknowledgments. C.H.N. is supported by JSPS. H.M. has been partially supported by BIRD, JST (Japan Science and Technology Agency). We acknowledge anonymous reviewers for helpful comments.

References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9, 1981–2014 (2008)
2. Ben-Hur, A., Noble, W.S.: Kernel methods for predicting protein-protein interactions. In: ISMB (Supplement of Bioinformatics), pp. 38–46 (2005)

3. Bleakley, K., Biau, G., Vert, J.-P.: Supervised reconstruction of biological networks with local models. In: ISMB/ECCB (Supplement of Bioinformatics), pp. 57–65 (2007)
4. Deng, M., Mehta, S., Sun, F., Chen, T.: Inferring domain-domain interactions from protein-protein interactions. In: Proceedings of the Sixth Annual International Conference on Computational Biology, pp. 117–126. ACM Press, New York (2002)
5. Ding, C.: Orthogonal nonnegative matrix tri-factorizations for clustering. In: SIGKDD, pp. 126–135 (2006)
6. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press, Cambridge (2007)
7. Kandola, J.S., Shawe-Taylor, J., Cristianini, N.: Learning semantic similarity. In: NIPS, pp. 657–664 (2002)
8. Kato, T., Tsuda, K., Asai, K.: Selective integration of multiple biological data for supervised network inference. Bioinformatics 21(10), 2488–2495 (2005)
9. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proceedings of the 21st National Conference on Artificial Intelligence, vol. 1, pp. 381–388. AAAI Press, Menlo Park (2006)
10. Kondor, R.I., Lafferty, J.D.: Diffusion kernels on graphs and other discrete input spaces. In: ICML, pp. 315–322 (2002)
11. Kunegis, J., Lommatzsch, A.: Learning spectral graph transformations for link prediction. In: ICML 2009: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 561–568. ACM, New York (2009)
12. Liben-Nowell, D., Kleinberg, J.M.: The link prediction problem for social networks. In: CIKM, pp. 556–559 (2003)
13. Miller, K., Griffiths, T., Jordan, M.: Nonparametric latent feature models for link prediction. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems, vol. 22, pp. 1276–1284 (2009)
14. Navarro, D.J., Griffiths, T.L.: A nonparametric bayesian method for inferring features from similarity judgments. In: NIPS, pp. 1033–1040 (2006)
15. Newman, M.E.J.: Clustering and preferential attachment in growing networks. Physical Review E 64(2), 025102+ (2001)
16. Shimbo, M., Ito, T., Mochihashi, D., Matsumoto, Y.: On the properties of von neumann kernels for link analysis. Machine Learning Journal 75(1), 37–67 (2009)
17. Smola, A.J., Kondor, R.: Kernels and regularization on graphs. In: Schölkopf, B., Warmuth, M.K. (eds.) COLT/Kernel 2003. LNCS (LNAI), vol. 2777, pp. 144–158. Springer, Heidelberg (2003)
18. Vert, J.-P., Yamanishi, Y.: Supervised graph inference. In: NIPS (2004)
19. Yamanishi, Y.: Supervised bipartite graph inference. In: NIPS, pp. 1841–1848 (2008)

Frequency-Aware Truncated Methods for Sparse Online Learning

Hidekazu Oiwa, Shin Matsushima, and Hiroshi Nakagawa

University of Tokyo,
7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan
{oiwa,masin}@r.dl.itc.u-tokyo.ac.jp,
n3@dl.itc.u-tokyo.ac.jp

Abstract. Online supervised learning with L_1-regularization has gained attention recently because it generally requires less computational time and a smaller space of complexity than batch-type learning methods. However, a simple L_1-regularization method used in an online setting has the side effect that rare features tend to be truncated more than necessary. In fact, feature frequency is highly skewed in many applications. We developed a new family of L_1-regularization methods based on the previous updates for loss minimization in linear online learning settings. Our methods can identify and retain low-frequency occurrence but informative features at the same computational cost and convergence rate as previous works. Moreover, we combined our methods with a cumulative penalty model to derive more robust models over noisy data. We applied our methods to several datasets and empirically evaluated the performance of our algorithms. Experimental results showed that our frequency-aware truncated models improved the prediction accuracy.

Keywords: Online Learning, L_1-regularization, Sparse Learning, Convex Programming, Low-frequency Occurrence Features, Natural Language Processing.

1 Introduction

Online learning is a training method using a sequence of instances, and it executes a learning process on one piece of data at each round. When learning from a large quantity of data, many well-known batch-type algorithms cannot solve an optimization problem within a reasonable time because the computational cost is very high. In addition, all instances may not be loaded into the main memory simultaneously. An online learning framework calculates what components of the weight vector are to be updated and by how much, based on only one instance, resulting in use of much less memory space. In this aspect, online learning is competitive for training from large-scale datasets in which the instances are high dimensional or the number of instances is very large. Online learning has recently attracted much attention owing to these properties, and many algorithms have been transformed into online ones.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 533–548, 2011.

L_1-regularization, Lasso, is regarded as a useful technique for large-scale data analysis. Normal L_1-regularization introduces the L_1 norm into optimization problems to penalize the weight vector. By applying L_1-regularization in algorithms, we can generate compact models to eliminate the features that do not contribute to the prediction. Compact models are also able to reduce the computational time and memory space used.

Carpenter[3] proposed an approach that combines online learning with L_1-regularization while maintaining the advantages of both techniques. Duchi et al.[7] and Langford et al.[9] generalized online learning with regularization and proved the regret bound. These methods consist of two steps. In the first step, the weight vector is updated to improve precision by reducing the value of the loss function using the received instance. Then, in the second step, regularization is applied to the weight vector. This learning scheme is the most famous in the field of sparse online learning. Therefore, many algorithms associated with the two-step scheme have been developed and analyzed, e.g., the lazy-update and cumulative-update forms. Thus, we focus on two-step scheme in this paper.

The widely known form of two-step algorithms is a subgradient method with L_1-regularization. This framework updates the weight vector in a loss minimization step according to the subgradient method, and then it truncates parameters using a normal L_1-regularized term. In this paper, we call this method SubGradient method with L_1-regularization (SG-L_1). Although SG-L_1 is an effective learning framework, this algorithm does not take into account feature frequency information. As a result, a set of rarely occurring features tends to be truncated to zero even if they are important or critical features. In many applications, such as natural language processing and pattern recognition tasks, the frequency of feature occurrence is not usually uniform. If there are value range differences among features, the truncated problem also occurs. However, these properties were not studied in detail in previous works.

Parts of infrequently occurring features are often informative for prediction. To capture these parts, pre-emphasizing methods have been developed, such as TF-IDF[14]. Another pre-processing method is to normalize the value range of each feature to standardize each feature. However, in an online learning setting, it is difficult to use these pre-processing methods while preserving the essence of online learning, i.e., to process samples sequentially.

In this paper, we propose simple truncated methods for retaining rarely occurring but informative features in an online setting. The key idea is to integrate the updating values in the loss minimization step into the L_1-regularization step. We call these methods frequency-aware truncated methods. In this way, we can decrease the truncation effects of rare features in an online setting. We also analyzed theoretical guarantees of our methods and derived the same computational cost and regret bound as for the SG-L_1 method. Furthermore, we investigated frequency-aware truncated methods with a cumulative penalty[15] to achieve robust solutions for noisy instances. We evaluated the effectiveness of our methods in experiments comparing our approach to other sparse online algorithms.

Table 1. Notation

a	scalar	$\lvert\lambda\rvert$	absolute value
$\mathbf{a}$	vector	$a^{(i)}$	i-th entry of vector $\mathbf{a}$
$\mathbf{A}$	matrix	$A^{(i,j)}$	(i,j)-th entry of matrix $\mathbf{A}$
$\lVert\mathbf{a}\rVert_p$	L_p norm	$\langle\mathbf{a},\mathbf{b}\rangle$	inner product

The outline of this paper is as follows. First, we introduce the problem setting and related works of sparse online learning in section 2. Next, we point out the disadvantages of previous works, namely, that low-frequency features are readily truncated, and propose frequency-aware truncated methods for solving rare-frequency feature truncated problems in section 3. Moreover, we analyze some properties of our methods and give theoretical guarantees. In section 4, we derive additional algorithms for combining our proposed methods with cumulative penalty models. In section 5, we evaluate the performance of our methods using classification tasks. From the experimental results, we discuss the properties of frequency-aware truncation and our contribution. We conclude the paper in section 6.

2 Linear Sparse Online Supervised Learning

2.1 Problem Setting

First, we introduce our notation to formally describe the problem setting. In this paper, scalars are lower-case italic letters, e.g., λ, and an absolute value of each scalar is $|\lambda|$. Vectors are lower-case bold letters, such as $\mathbf{x}$. Matrices are upper-case bold letters, e.g., $\mathbf{X}$. $\|\mathbf{x}\|_p$ represents L_p norm of vector $\mathbf{x}$, and $\langle\mathbf{x},\mathbf{y}\rangle$ denotes an inner product of two vectors $\mathbf{x},\mathbf{y}$. Table 1 summarizes the notation in this paper.

In this work, we develop a new family of truncated strategies for linear online learning. In the setting of standard linear sparse online learning, algorithms perform a sequential prediction and updating scheme. The objective is to derive the optimal weight vector $\mathbf{w} \in W \subset \mathbf{R}^d$, where W is a closed convex set. The updating process is conducted as follows.

1. Receive input data $\mathbf{x}_t \in X \subset \mathbf{R}^d$. Input data $\mathbf{x}_t$ is a feature vector taken from a d-dimensional closed convex set X.
2. Make a prediction based on an inner product of feature vector $\mathbf{x}_t$ and a weight vector $\mathbf{w}_t$. The predicted value is $\hat{y}_t = \langle\mathbf{w}_t,\mathbf{x}_t\rangle$.
3. Observe a true output y_t.
4. Update weight vector $\mathbf{w}_t$ to $\mathbf{w}_{t+1/2}$ using a loss function $\ell_t(\cdot)$.
5. Update $\mathbf{w}_{t+1/2}$ to $\mathbf{w}_{t+1}$ using an L_1-regularized term $r_t(\cdot)$.
6. Iterate steps 1 through 5 until no input data remains.

We update a weight vector according to loss function ℓ_t at step 4 and regularization term r_t at step 5. ℓ_t is a loss function of the form[1]

$$\ell_t(\mathbf{w}_t) : W \to \mathbf{R}_+ \ ,$$

where ℓ_t is convex with respect to weight vector $\mathbf{w}_t$. In this paper, we deal with linear online learning framework. Thus, we consider a loss function that exists a function $\hat{\ell}_t$, where

$$\ell_t(\mathbf{w}) = \hat{\ell}_t(\langle \mathbf{w}, \mathbf{x}_t \rangle) = \hat{\ell}_t(\hat{y}_t) \ , \tag{1}$$

and loss function $\hat{\ell}_t$ is generally non-decreasing for the difference between $\hat{y}_t$ and y_t. We call a loss function that satisfies the restriction above a linear online learning problem.

In the setting of a standard linear online learning problem, a subgradient method(SG) [1, 17] is often used for learning. In subgradient methods, the weight vector is updated as stated in formula (2):

$$\mathbf{w}_{t+1/2} = \mathbf{w}_t - \eta_t \mathbf{g}_t^f \quad s.t. \quad \mathbf{g}_t^f \in \partial f_t(\mathbf{w}_t) \ , \tag{2}$$

where $\mathbf{g}_t^f$ is a subgradient[2] of f_t with respect to $\mathbf{w}_t$ and η_t is a learning rate. $\partial f(\mathbf{w}_t)$ is a set of all subgradients of f_t at $\mathbf{w}_t$. A subgradient method updates parameters sequentially to minimize f_t. It has been proved that the regret bound of the subgradient method is $O(\sqrt{T})$ when $\eta_t = 1/\sqrt{t}$, and consequently the regret bound per data vanishes as $T \to \infty$.

r_t is a regularized term of the form

$$r_t(\mathbf{w}_{t+1/2}) : W \to \mathbf{R}_+ \ ,$$

where r_t is convex in $\mathbf{w}_{t+1/2}$. Many algorithms use L_1 norm to penalize the weight vector in a sparsity-induced regularization.

$$r_t(\mathbf{w}) = r(\mathbf{w}) = \lambda \|\mathbf{w}\|_1 \ , \tag{3}$$

where λ is a regularization parameter.

In SG-L_1, we penalize the weight vector according to formula (4) at step 5.

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \left\{ \|\mathbf{w} - \mathbf{w}_{t+1/2}\|_2^2/2 + \lambda \eta_{t+1/2} \|\mathbf{w}\|_1 \right\} \ , \tag{4}$$

where $\eta_{t+1/2}$ is the second learning rate. In this step, we find a weight vector that is in between the previous weight $\mathbf{w}_{t+1/2}$ and a truncated one.

In this paper, we focus on step 5, the regularization step, and propose a new family of L_1-regularization methods.

[1] Squared loss function $\ell_t(\mathbf{w}_t) = (y_t - \langle \mathbf{w}_t, \mathbf{x}_t \rangle)^2$ and Hinge loss function $\ell_t(\mathbf{w}_t) = [1 - y_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle]_+$ are usually used for ℓ_t.

[2] A subgradient of f at $\mathbf{x}$ is the vector $\mathbf{g} \in \mathbf{R}^n$ that satisfies

$$\forall \mathbf{y} \quad f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \mathbf{g}, \mathbf{y} - \mathbf{x} \rangle \ .$$

Even if f is non-differentiable, at least one subgradient exists when f is convex.

2.2 Related Works

As previously mentioned, SG-L_1 is the most common method in sparse online learning frameworks. Carpenter[3] split the update procedure into two steps and proposed a method to obtain a sparse solution in an online setting. In addition, FOBOS[7] and truncated gradient methods[9] generalized a splitting form method and analyzed the optimal step size and the regret bound of sparse online learning. These algorithms are guaranteed to asymptotically offer regret $O(\sqrt{T})$ in the restriction of the loss function and regularized term in section 2.1.

Furthermore, Nesterov[12] proposed the dual averaging method for online learning. This method updates the weight vector to solve the simple optimization problem that includes the average of all previous subgradients of the loss functions at each iteration. Xiao[16] developed the extension of the dual averaging method to include a regularization term, such as L_1 norm. The regularized dual averaging form (RDA) solves the minimization problem that takes into account both a regularized term and the average of all previous subgradients. A family of dual averaging methods ensures the $O(\sqrt{T})$ regret bound, but, this scheme also has the low-occurrence feature truncation problem because it applies the same penalty to all features. Duchi et al.[6] proposed a new family of subgradient methods as an alternative to previously used subgradient methods, named AdaGrad. AdaGrad incorporates the knowledge of the data observed in earlier iterations to emphasize the infrequently occurring instance in an online setting. However, when a feature occurs for the first time, AdaGrad cannot standardize it. This is because AdaGrad adjusts the update in a loss minimization step. In addition, AdaGrad also has the value range problem explained in section 3.

Useful methods have been proposed in the field of online learning for classification. For example, Perceptron[13], Passive-Aggressive[4], and Confidence-Weighted[5] algorithms are often used as alternatives to subgradient methods. In particular, Confidence-Weighted algorithms introduce a Gaussian distribution into a weight vector and update parameters using the covariance parameters of the weight vector in order to emphasize informative low-frequency features. However, Confidence-Weighted algorithms do not generate a sparse solution.

3 Frequency-Aware Truncated Methods

As noted in section 1, SG-L_1 and other sparse online algorithms apply the same penalty to all features independent of the previous update of each feature. In the linear online learning framework, algorithms update the weight of the feature that occurs in a piece of input data. As a result, the set of rarely occurring features tends to be sparse because the value range of these parameters must be larger than that of other features that are not truncated.

For example, we apply SG-L_1 to the dataset in which feature A's occurrence rate is 1/2 and feature B's rate is 1/100. In this case, feature B inevitably becomes 0 unless feature B's update satisfies

$$\eta_t |g_t^{\ell,(B)}| \geq \lambda \sum_{s=t}^{t+100} \eta_{s+1/2} \ .$$

In this paper, $g^{(i)}$ represents the i-th entry of the vector $\mathbf{g}$. On another front, the weight of feature A does not always drop to 0, where

$$\eta_t |g_t^{\ell,(A)}| \geq \lambda \sum_{s=t}^{t+1} \eta_{s+1/2} \ .$$

Therefore, in a normal sparse online learning framework, if the feature occurrence rate is non-uniform, we may fail to retain rarely occurring but important features. In many tasks, such as NLP and pattern recognition, a feature's occurrence rate is usually non-uniform.

In addition, each feature's value range affects the truncation of parameters. Assume that there are two features: one is an arbitrary feature and the other is one whose value is 1000 times larger than the first feature. If we learn from this dataset using normal sparse online learning, which applies the same penalty to all features, the weight of the first feature is truncated faster than that of the second feature, despite them both having almost the same effect for prediction.

We designed a family of frequency-aware truncated methods to capture low-frequency features and solve the value range problem in an online setting. A frequency-aware truncated method redefines step 5 in a sparse online learning alternative to normal L_1 norm by using each feature's previous update.

Let $\mathbf{u}_t$ be the t-th update at step 4[3]. In this case, we can write step 4 as

$$\mathbf{w}_{t+1/2} = \mathbf{w}_t + \mathbf{u}_t \ .$$

Then, the frequency-aware truncated method defines step 5 as follows:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \left\{ \|\mathbf{w} - \mathbf{w}_{t+1/2}\|_2^2/2 + \lambda \eta_{t+1/2} \|\mathbf{H}_{t,p}\mathbf{w}\|_1 \right\} \ , \tag{5}$$

where

$$\mathbf{H}_{t,p} = \begin{pmatrix} h_{t,p}^{(1)} & 0 & \dots & 0 \\ 0 & h_{t,p}^{(2)} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & h_{t,p}^{(d)} \end{pmatrix} \quad s.t. \quad h_{t,p}^{(j)} = \sqrt[p]{\sum_{s=1}^{t} \left| u_s^{(j)} \right|^p} \ .$$

$h_{t,p}^{(j)}$, which represents frequency-awareness, is the L_p norm of a vector that consists of feature j's update at step 4 in each iteration. $\mathbf{H}_{t,p}$ is a matrix consisting of $h_{t,p}^{(j)}$ of all features in a diagonal component. In this definition, we can derive the vector in which each component is $h_{t,p}^{(j)} w_t^{(j)}$, or $\mathbf{H}_{t,p}\mathbf{w}_t$. Thus, from equation (5), vector component $w_t^{(j)}$ tends to be truncated when the value of $h_{t,p}^{(j)}$ is large. If a feature is rarely occurring, the number of updates is also small; thus, $h_{t,p}^{(j)}$ also tends to have a small value. In addition, if the value of a feature C is 1000

[3] The update value of this form can be obtained in the setting of linear online learning (e.g., $-\eta_t \mathbf{g}_t^{\ell}$ in a subgradient method).

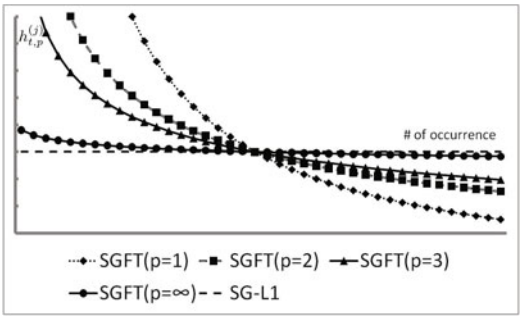


Fig. 1. Comparison of awareness parameter $h_{t,p}^{(j)}$ against parameter p

times larger than that of a feature D, then, $h_{t,p}^{(C)}$ is also 1000 times larger than $h_{t,p}^{(D)}$. Thus, we can keep the truncation of these two features the same in effect.

We can set a wide variety of numbers into parameter p to adjust the importance of rare features. To show how an awareness paramter $h_{t,p}^{(j)}$ is influenced by parameter p, we assume a simple example, in which gradient's value is limited to either 0 or 1, and represent the relationship between the value of $h_{t,p}^{(j)}$ and the count of feature j's occurrence. The example is illustrated in Fig. 1. A horizontal plot describes the count of occurrence in descending order and a vertical plot shows the value of $h_{t,p}^{(j)}$. It indicates that the smaller the value of p, the more slowly a rare feature is truncated. We note that a normal L_1 can be regarded as the algorithm of $h_{t,p}^{(j)} = 1$ for all t, j.

3.1 Subgradient Method with Frequency-Aware Truncation

In the following sections, we focus on the SubGradient method with Frequency-aware Truncation, which we call SGFT.

In SGFT, we can derive the update function as follows:

$$\begin{aligned} w_{t+1}^{(j)} &= sign\left(w_{t+1/2}^{(j)}\right)\left[\left|w_{t+1/2}^{(j)}\right| - \eta_{t+1/2} h_{t,p}^{(j)} \lambda\right]_+ \\ &= sign\left(w_t^{(j)} - \eta_t g_t^{\ell,(j)}\right)\left[\left|w_t^{(j)} - \eta_t g_t^{\ell,(j)}\right| - \eta_{t+1/2} h_{t,p}^{(j)} \lambda\right]_+ . \end{aligned} \tag{6}$$

The process of deriving this updating function is the same as that by Duchi et al. [7]. Equation (6) shows that SGFT can process one piece of data at $O(d)$ computational cost as large as SG-L_1. Fig. 2 illustrates how $h_{t,p}^{(j)}$ affects the updating of $\mathbf{w}_{t+1/2}$ in the regularization step. These figures indicate that the parameter $h_{t,p}^{(j)}$ adjusts the intensity of truncation to retain rarely occurring but informative features.

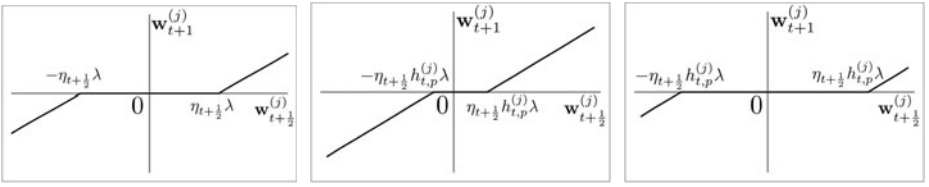


Fig. 2. These figures show how $h_{t,p}^{(j)}$ affects the regularization. Left : Normal SG-L_1 case, Center : Small $h_{t,p}^{(j)}$ case in SGFT, Right : Large $h_{t,p}^{(j)}$ case in SGFT.

3.2 Regret Analysis of SGFT

In SGFT, regularization term r_t is replaced with $r_t(\mathbf{w}_t) = \lambda\|\mathbf{H}_{t,p}\mathbf{w}_t\|_1$ from a normal L_1 norm $r(\mathbf{w}_t) = \lambda\|\mathbf{w}_t\|_1$. When differentiating r_t with respect to a weight vector $\mathbf{w}$ and applying L_2 norm, we obtain

$$\|\partial r_t\|_2 = \lambda\sqrt{\sum_{k=1}^{d}\left(h_{t,p}^{(k)}\right)^2}\,. \tag{7}$$

From equation (7), Lemma 1 is proved.

Lemma 1. *We define* $\|\partial f\|$ *as* $\sup_{\mathbf{g}\in\partial f(\mathbf{w})}\|\mathbf{g}\|_2$. *If* $\|\partial\ell_t\| \le G$, $\eta_t = \eta_{t+1/2} = c/\sqrt{t}$ *using a scalar* $c > 0$, *and* $h_{t,p}^{(k)}$ *is* L_p *norm where* $p > 2$, *a scalar* U *exists that satisfies inequality (8).*

$$\lim_{t\to\infty}\|\partial r_t\| < U\,. \tag{8}$$

In the Appendix, we prove formula (8).

In the case of $p \le 2$, we redefine the diagonal matrix $\mathbf{H}_t$ as $H_t^{(k,k)} = \min(h_{t,p}^{(k)}, V)$ using a scalar V. In this paper, $H^{(i,j)}$ represents the (i,j)-th entry of the matrix $\mathbf{H}$. In the case of $p \le 2$, we can prove that the upper bound of $\|\partial r_t\|$ is $\sqrt{d}\lambda V$. Thus, there is a scalar U where $\lim_{t\to\infty}\|\partial r_t\| \le U$. In this case, we can prove that the regret bound of the SGFT is $O(\sqrt{T})$. The proof is in the Appendix.

Theorem 1. *We define the matrix* $\mathbf{H}_{t,p}$ *as*

$$H_{t,p}^{(k,k)} = \begin{cases} \min(h_{t,p}^{(k)}, V) & p \le 2 \\ h_{t,p}^{(k)} & p > 2 \end{cases} \tag{9}$$

In addition, assume both the loss function and regularization term are convex functions, and that they satisfy $\forall\mathbf{w}_t \quad \|\mathbf{w}_t - \mathbf{w}^*\|_2 \le D$, $\|\partial\ell_t\| \le U$, $\|\partial r_t\| \le U$ *where setting* $\eta_t = \eta_{t+1/2} = c/\sqrt{t}$ *using scalars* D, U, *and* $c > 0$.

In this case, the regret bound of SGFT satisfies formula (10).

$$R_{\ell+r}(T) \le 2UD + \left(D^2/2c + 8U^2c\right)\sqrt{T} = O(\sqrt{T})\,, \tag{10}$$

where

$$R_{\ell+r}(T) = \sum_{t=1}^{T} \{\ell_t(\mathbf{w}_t) + r_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*) - r_t(\mathbf{w}^*)\},$$

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_{t=1}^{T} \{\ell_t(\mathbf{w}) + r_t(\mathbf{w})\}.$$

3.3 Lazy Update

SGFT allows us to truncate parameters in a lazy fashion. We do not need to penalize the weights of features that do not occur in the current sample, thus, we can postpone applying the penalty at each iteration. This updating scheme enables faster calculation when the dimension of instances is large and we have sparse samples.

We define the absolute value of the total L_1 penalty from $t = 1$ to n as u_n.

$$u_n = \lambda \sum_{t=1}^{n} \eta_{t+1/2}. \tag{11}$$

At each instance, before we update the parameter in step 4, we apply the L_1 penalty to features that are used in the input.

$$w_{t+1/2}^{(j)} = \begin{cases} \max\left(0, w_t^{(j)} - (u_{t-1} - u_{s-1})h_{s,p}^{(j)}\right) & w_t^{(j)} \geq 0 \\ \min\left(0, w_t^{(j)} + (u_{t-1} - u_{s-1})h_{s,p}^{(j)}\right) & w_t^{(j)} < 0 \end{cases}, \tag{12}$$

where s is the sample number that feature j is used at the end. If the value of u_{s-1} is calculated at s-th update, which is the last update of the weight of feature j, only u_t, u_{t-1} must be derived at iteration t and the value from u_1 to u_{t-2} does not have to be preserved.

Second, we perform a subgradient method using $\mathbf{w}_{t+1/2}$ in step 4 and derive $\mathbf{w}_{t+1}$. Finally, we skip step 5 to finish. In the lazy update version of SGFT, we can compute the update at the speed of O(number of features that occur).

4 SGFT with Cumulative Penalty

Tsuruoka et al.[15] proposed a cumulative penalty model for SG-L_1. The normal SG-L_1 has a problem where a solution is often obtained that is significantly affected by the last few instances. This is because the weight easily moves away from zero when a feature is used in the last few instances. The main idea of the cumulative penalty model is to keep track of the total penalty. Then, we apply a cumulative L_1 penalty to smooth the effect of the update fluctuation and move away from zero unless the updating sum exceeds the cumulative penalty. In addition, this model can smooth the effect of the update and also suppress

noisy data. In this section, we propose a method combining our models with the cumulative penalty model.

We begin by introducing $q_t^{(j)}$ as an already applied cumulative L_1 penalty of feature j at the t-th instance. We initialize $q_0^{(j)} = 0$ for all j. In this setting, at step 5, we update the weight vector whose feature is used in the current instance as follows:

$$w_{t+1}^{(j)} = \begin{cases} \max\left(0, w_{t+1/2}^{(j)} - (h_{t,p}^{(j)} u_t + q_t^{(j)})\right) & w_{t+1/2}^{(j)} \geq 0 \\ \min\left(0, w_{t+1/2}^{(j)} + (h_{t,p}^{(j)} u_t - q_t^{(j)})\right) & w_{t+1/2}^{(j)} < 0 \end{cases} . \tag{13}$$

Then, we update the parameter $q_t^{(j)}$ if the feature j is used in the current instance as follows:

$$q_t^{(j)} = q_{t-1}^{(j)} + (w_{t+1}^{(j)} - w_{t+1/2}^{(j)}) . \tag{14}$$

In a cumulative penalty setting, we rewrite the optimization problem as if returning to the previous iteration and applying the new frequency-aware adaptation parameter $h_{t,p}^{(j)}$. This reformalization makes the update function simple and reduce space complexity. The same as Tsuruoka et al. did, the whole frequency-aware L_1 penalty is applied at once if the following two types of weight vectors reside within the same orthant: 1) the weight vector that had been updated by the true gradient with the latest penalty and 2) the weight vector calculated with the cumulative form of L_1 normalization.

SGFT with cumulative penalty takes O(number of features that occur) computational time at each iteration.

5 Evaluation

We evaluated our proposed frequency-aware truncated methods using classification tasks. In the experiment, we used three datasets.

First, we used sentiment classification tasks[2] for Amazon.com goods reviews. Classification tasks classify whether a positive or negative opinion is noted in each review. In this dataset, we used the books and dvd categories.

Second, we used the 20 Newsgroups dataset (news20)[8]. The news20 is a news categorization task in which a learning algorithm predicts to what category each news article is assigned. This dataset consists of about 20,000 news articles. Each article is assigned to one of 20 predetermined categories. We used four subsets of news20: ob-2-1, sb-2-1, ob-8-1, and sb-8-1[11]. In each subset, the number of categories and the closeness among categories differed. For the first letter of each subset name, 'o' indicates 'overlap' and 's' denotes 'separated'. Classifying categories correctly is more difficult with an 'overlap' dataset. The second letter of the subset names means the heterogeneity among the categories and there is no difference in the instance number among the categories. The middle number is the number of categories.

Last, we used the Reuters-21578 [10] dataset. The Reuters-21578 also consists of news articles and we used a dataset for a 20-category classification task

Table 2. Dataset specifications

	# of instances	# of features	# of categories
books	4,465	332,441	2
dvd	3,586	282,901	2
ob-2-1	1,000	5,942	2
sb-2-1	1,000	6,276	2
ob-8-1	4,000	13,890	8
sb-8-1	4,000	16,282	8
reut20	7,800	34,488	20

(reut20) from the Reuters-21578. In Table 2, we provide the specifications of each dataset, including the number of features, instances, and categories.

In this experiment, we used the hinge-loss function as a loss function. When there are more than two categories, it is not possible to use hinge-loss directly because the hinge-loss function was developed for binary categorization. In our experiment, we defined a weight vector as $\mathbf{w} \in \hat{W} \subset \mathbf{R}^{d \times K}$, where K was the number of classes, and a feature vector as $\Phi(\mathbf{x}, y)$, mapped from the Cartesian product $X \times Y$, where Y was the set of labels in the 1-of-K scheme. Moreover, we set the loss function as formula (15).

$$\ell_t(\mathbf{w}_t) = [1 - \langle \mathbf{w}_t, \Phi(\mathbf{x}_t, y_t) \rangle + \max_{z_t \in Y \setminus y_t} \langle \mathbf{w}_t, \Phi(\mathbf{x}_t, z_t) \rangle]_+ , \tag{15}$$

where y_t is a correct label at t. We can process the multi-class classification tasks as define above. In the experiment, we examined SGFT, SG-L_1, and RDA[16] to compare the precision and sparseness rates. From a family of frequency-aware truncated methods, we selected the algorithms of $p = 1, 2, 3, \infty$.

The step size η_t was set at $\eta_t = \eta_{t+1/2} = 1/\sqrt{t}$ to satisfy the restriction of the regret bound in SGFT and SG-L_1. Moreover, in SGFT where $p = 1, 2$, we set $V = 500$ to satisfy the regret bound restriction[4]. In contrast, in RDA, we set $h(\mathbf{w}) = 1/2\|\mathbf{w}\|_2^2$ and $\beta_t = \sqrt{t}$. In our experiment, we evaluated the performance of our methods using a ten-fold cross-validation to achieve the highest precision rate by adjusting the parameter λ. We set the number of iterations to 20.

The experimental results of SGFT against the change of parameter p are shown in Table 3. The figure in $[\cdot]$ means the standard deviation, and the figure in $(\cdot)$ denotes the sparseness rate. Moreover, the highest precision rates among all the algorithms are written in **bold** font.

Table 3 indicates that SGFT with $p = 2$ achieves the best performance in the four datasets. Moreover, in the other three datasets, SGFT $p = 2$ has the second highest precision, indicating SGFT $p = 2$ is an efficient learning method in the SGFT family. Table 3 also shows that SGFT has a tendency of increasing sparsity responding to increase of parameter p.

[4] In this experiment, the value of $h_{t,p}^{(j)}$ did not exceed 500, thus the value of V did not influence the result.

Table 3. SGFT's precision (sparseness) rate against parameter p (Iterations : 20)

	SGFT ($p = 1$)	SGFT ($p = 2$)	SGFT ($p = 3$)	SGFT ($p = \infty$)
books	85.23[1.52] (34.52)	**85.52**[1.24] (48.26)	85.14[1.33] (49.58)	85.05[1.41] (69.39)
dvd	82.49[1.68] (37.46)	84.75[1.75] (59.72)	**85.03**[2.28] (63.74)	84.02[1.66] (67.19)
ob-2-1	97.00[1.73] (42.78)	**97.10**[1.14] (56.73)	96.90[1.87] (59.03)	96.80[1.94] (59.78)
sb-2-1	**98.90**[0.83] (60.13)	98.40[0.80] (70.32)	98.40[1.11] (71.99)	98.10[1.14] (72.69)
ob-8-1	92.25[1.14] (62.83)	**93.10**[1.41] (62.84)	93.00[1.29] (64.64)	91.45[1.33] (77.78)
sb-8-1	90.90[1.72] (68.26)	92.55[1.85] (68.49)	**93.78**[2.44] (70.23)	91.25[1.44] (83.53)
reut20	95.23[0.65] (89.11)	**96.04**[0.56] (90.38)	95.91[0.55] (90.21)	94.80[0.67] (91.05)

Table 4. Precision (sparseness) rate (Iterations : 20)

	SGFT ($p = 2$)	SG-L_1	RDA
books	85.52[1.24] (48.26)	84.98[1.61] (48.28)	**86.57**[1.16] (34.65)
dvd	84.75[1.75] (59.72)	83.91[1.55] (79.57)	**86.36**[2.08] (37.08)
ob-2-1	97.10[1.14] (56.73)	96.40[1.96] (49.23)	**97.60**[1.80] (39.83)
sb-2-1	**98.40**[0.80] (70.32)	97.20[1.78] (84.25)	98.20[0.75] (56.67)
ob-8-1	93.10[1.41] (62.84)	90.63[1.64] (87.90)	**93.78**[1.21] (50.52)
sb-8-1	92.55[1.85] (68.49)	90.53[1.61] (67.46)	**95.45**[0.95] (60.46)
reut20	96.04[0.56] (90.38)	95.53[0.63] (89.29)	**96.27**[0.63] (86.67)

Table 4 illustrates the results of SG-L_1 and RDA as compared with SGFT $p = 2$, which showed the most efficient performance in Table 3.

From Table 4, SGFT is confirmed to outperform SG-L_1 in all the datasets. At the same time, SGFT does not necessarily have a smaller sparsity rate than SG-L_1. This result shows that frequency-aware truncation improves the accuracy of precision, without degrading sparsity. From the experimental results, note that frequency-aware truncation could improve the accuracy by retaining rarely occurring but important features and dropping unimportant features. Thus, in the setting of sparse online learning, frequency-aware truncation is a useful method compared with the normal L_1-regularization for these datasets.

We also evaluated the experimental results of RDA. The results showed that RDA obtained the highest precision rate in these tasks except for sb-2-1, but, the rate of sparsity was smaller than SGFT. This indicates that RDA is a sophisticated algorithm for precise learning; however, to obtain a sparse solution, frequency-aware truncation methods are also efficient for learning. We consider that the margin between these two methods occur partly because RDA[16] has the smaller regret bound than FOBOS[7] and SGFT in terms of the coefficient.

6 Conclusion

We analyzed a new family of truncated methods for retaining rarely occurring features in an online setting. These methods integrate the sum of updates in the loss minimization steps into the regularization step to adjust the intensity of truncation. In this way, we can solve the problem where rarely used features

are truncated on a priority basis. Specifically, we proved the computational cost and theoretical guarantees of SGFT, which is also known as a frequency-aware truncated method. In addition, we provided possible extensions of our work, such as lazy-update and cumulative-penalty schemes. Finally, we evaluated the performance of our methods in experiments. The experimental results showed that frequency-aware truncated methods could retain rarely occurring but important features without loss of sparsity.

A few discussions for further research remain in connection with our proposed methods. The first is the integration of frequency-aware methods into a primal-dual averaging framework. We assume that a frequency-aware scheme could be connected with dual-averaging methods with a minor change of frequency-aware term's definition. This extension would also enable us to give the same regret bound and computational time as those for dual-averaging methods and expect the higher performance than RDA. The second issue is whether we can optimize parameter p in an online setting. We aim to investigate these questions and further extensions of our proposed methods.

References

1. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific (1999)
2. Blitzer, J., Dredze, M., Pereira, F.: Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In: Association for Computational Linguistics (ACL), pp. 440–447 (2007), http://www.cs.jhu.edu/~mdredze/datasets/sentiment
3. Carpenter, B.: Lazy sparse stochastic gradient descent for regularized multinomial logistic regression. Technical report, Alias-i (2008)
4. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online Passive-Aggressive Algorithms. Journal of Machine Learning Research 7, 551–585 (2006)
5. Dredze, M., Crammer, K.: Confidence-weighted linear classification. In: ICML, pp. 264–271 (2008)
6. Duchi, J., Hazan, E., Singer, Y.: Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. In: COLT, pp. 257–269 (2010)
7. Duchi, J., Singer, Y.: Efficient Online and Batch Learning Using Forward Backward Splitting. Journal of Machine Learning Research 10, 2899–2934 (2009)
8. Lang, K.: Newsweeder: Learning to filter netnews. In: International Conference on Machine Learning (ICML), pp. 331–339 (1995), http://mlg.ucd.ie/datasets
9. Langford, J., Li, L., Zhang, T.: Sparse Online Learning via Truncated Gradient. J. Mach. Learn. Res. 10, 777–801 (2009)
10. Lewis, D.D.: Reuters-21578, http://www.daviddlewis.com/resources/testcollections/reuters21578
11. Matsushima, S., Shimizu, N., Yoshida, K., Ninomiya, T., Nakagawa, H.: Exact Passive-Aggressive Algorithm for Multiclass Classification Using Support Class. In: SDM, pp. 303–314 (2010)
12. Nesterov, F.: Primal-Dual subgradient methods for convex problems. Mathematical Programming 120(1), 221–259 (2009)
13. Rosenblatt, F.: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. Psychological Review 65, 386–408 (1958)

14. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
15. Tsuruoka, Y., Tsujii, J., Ananiadou, S.: Stochastic Gradient Descent Training for L1-regularized Log-linear. In: ACL-IJCNLP, pp. 477–485 (2009)
16. Xiao, L.: Dual averaging methods for regularized stochastic learning and online optimization. In: Advances in Neural Information Processing Systems, vol. 23 (2009)
17. Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: International Conference on Machine Learning (ICML), pp. 928–936 (2003)

Appendix

Proof of Lemma 1

Let $\boldsymbol{\eta}_t$ be a vector of $(\eta_1, \eta_2, \ldots, \eta_t)$. If $\|\partial \ell_t\| \leq G$ for all t, we can derive

$$h_{t,p}^{(k)} \leq \|G\boldsymbol{\eta}_t\|_p = G\|\boldsymbol{\eta}_t\|_p \ . \tag{16}$$

The first inequality follows from the inequality below.

$$\forall t, k \quad |g_t^{\ell,(k)}| \leq \|\mathbf{g}_t^{\ell}\|_2 \leq \|\partial \ell_t\| \leq G \ .$$

From the definition of L_p norm, we can rewrite equation (16) as

$$\|\boldsymbol{\eta}_t\|_p = \sqrt[p]{\sum_{k=1}^{t} |\eta_k|^p} \ . \tag{17}$$

Thus, if we substitute η_k with $c/\sqrt{k}$, we obtain equation (18).

$$\|\boldsymbol{\eta}_t\|_p = c\sqrt[p]{\sum_{k=1}^{t} k^{-\frac{p}{2}}} \ . \tag{18}$$

$\sum_{t=1}^{T} t^{-\frac{p}{2}}$ is a zeta function. From the characteristics of zeta functions, if $-\frac{p}{2} < -1$, that is, $p > 2$, $\sum_{k=1}^{t} k^{-\frac{p}{2}}$ has a upper bound and thus converges as $t \to \infty$. We set the upper limit value to S, obtaining $\|\boldsymbol{\eta}_t\|_p = cS^{\frac{1}{p}}$. Then, there is a scalar U which satisfies equation (19).

$$\|\partial r_t(\mathbf{w})\| \leq \lambda G\sqrt{\sum_{l=1}^{d} (cS^{\frac{1}{p}})^2} = c\lambda G S^{\frac{1}{p}}\sqrt{d} \leq U \ . \tag{19}$$

Therefore, we can prove Lemma 1. However, in the case of $p \leq 2$, we cannot bound $h_{t,p}^{(k)}$ because the zeta function does not converge.

Proof of Theorem 1

The procedure of the proof is similar to that by Duchi et al.[7], but, there is a small difference because, in our methods, the regularization term depends on t which is the number of iterations. First, we prove Lemma 2.

Lemma 2. *Assume both loss function ℓ_t and regularization term r_t have convexity and satisfy equation (20).*

$$\|\partial \ell_t(\mathbf{w})\|^2 \leq G^2, \|\partial r_t(\mathbf{w})\|^2 \leq G^2 \ . \tag{20}$$

Let step size η_t satisfy $\eta_{t+1} \leq \eta_{t+1/2} \leq \eta_t$ and $\eta_t \leq 2\eta_{t+1}$. In this case, we can prove equation (21).

$$\begin{aligned} \forall \mathbf{w}^* \quad \exists c \leq 5 \qquad & 2\eta_t \ell_t(\mathbf{w}_t) - 2\eta_t \ell_t(\mathbf{w}^*) + 2\eta_{t+1/2} r_t(\mathbf{w}_{t+1}) - 2\eta_{t+1/2} r_t(\mathbf{w}^*) \\ & \leq \|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2^2 + 8\eta_t^2 G^2 \ . \end{aligned} \tag{21}$$

From the condition that the loss function is convex, we can derive equation (22) in terms of any subgradient $\mathbf{g}_t^\ell \in \partial \ell_t(\mathbf{w}_t)$.

$$\ell_t(\mathbf{w}^*) \geq \ell_t(\mathbf{w}_t) + \langle \mathbf{g}_t^\ell, \mathbf{w}^* - \mathbf{w}_t \rangle \implies -\langle \mathbf{g}_t^\ell, \mathbf{w}_t - \mathbf{w}^* \rangle \leq \ell_t(\mathbf{w}^*) - \ell_t(\mathbf{w}_t) \ . \tag{22}$$

This is the case with regard to regularization term $r_t(\cdot)$. In this paper, we denote any subgradient of regularization term $r_t(\mathbf{w}_{t+1})$ as $\mathbf{g}_{t+1}^r$.

From the Cauchy-Shwartz inequality and equation (2), we obtain

$$\begin{aligned} \langle \mathbf{g}_{t+1}^r, \mathbf{w}_{t+1} - \mathbf{w}_t \rangle &= \langle \mathbf{g}_{t+1}^r, -\eta_t \mathbf{g}_t^\ell - \eta_{t+1/2} \mathbf{g}_{t+1}^r \rangle \\ &\leq \|\mathbf{g}_{t+1}^r\|_2 \|\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2} \mathbf{g}_{t+1}^r\|_2 \\ &\leq \eta_{t+1/2} \|\mathbf{g}_{t+1}^r\|_2^2 + \eta_t \|\mathbf{g}_{t+1}^r\|_2 \|\mathbf{g}_t^\ell\|_2 \\ &\leq (\eta_{t+1/2} + \eta_t) G^2 \ . \end{aligned} \tag{23}$$

In the first equation above, we use $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t^\ell - \eta_{t+1/2} \mathbf{g}_{t+1}^r$ derived from the derivation of equations (2) and (5).

Then, we proceed to derive the upper bound of the difference between $\mathbf{w}^*$ and $\mathbf{w}_{t+1}$ for obtaining the upper bound of $\ell_t(\mathbf{w}_t) + r_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*) - r_t(\mathbf{w}^*)$. We can expand the L_2 norm of the difference between $\mathbf{w}^*$ and $\mathbf{w}_{t+1}$ as follows:

$$\begin{aligned} \|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2^2 &= \|\mathbf{w}_t - (\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2} \mathbf{g}_{t+1}^r) - \mathbf{w}^*\|_2^2 \\ &= \|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - 2\left(\eta_t \langle \mathbf{g}_t^\ell, \mathbf{w}_t - \mathbf{w}^* \rangle + \eta_{t+1/2} \langle \mathbf{g}_{t+1}^r, \mathbf{w}_t - \mathbf{w}^* \rangle\right) \\ &\quad + \|\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2} \mathbf{g}_{t+1}^r\|_2^2 \\ &= \|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - 2\eta_t \langle \mathbf{g}_t^\ell, \mathbf{w}_t - \mathbf{w}^* \rangle + \|\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2} \mathbf{g}_{t+1}^r\|_2^2 \\ &\quad - 2\eta_{t+1/2}\left(\langle \mathbf{g}_{t+1}^r, \mathbf{w}_{t+1} - \mathbf{w}^* \rangle - \langle \mathbf{g}_{t+1}^r, \mathbf{w}_{t+1} - \mathbf{w}_t \rangle\right) \ . \end{aligned} \tag{24}$$

The bound of the third term is derived as

$$\begin{aligned} \|\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2} \mathbf{g}_{t+1}^r\|_2^2 &= \eta_t^2 \|\mathbf{g}_t^\ell\|_2^2 + 2\eta_t \eta_{t+1/2} \langle \mathbf{g}_t^\ell, \mathbf{g}_{t+1}^r \rangle + \eta_{t+1/2}^2 \|\mathbf{g}_{t+1}^r\|_2^2 \\ &\leq 4\eta_t^2 G^2 \ . \end{aligned} \tag{25}$$

The upper bound of equation (24) is obtained by equations (22), (23), and (25).

$$\begin{aligned}\|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2^2 &\leq \|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - 2\eta_t\langle \mathbf{g}_t^\ell, \mathbf{w}_t - \mathbf{w}^*\rangle - 2\eta_{t+1/2}\langle \mathbf{g}_{t+1}^r, \mathbf{w}_{t+1} - \mathbf{w}^*\rangle \\ &\quad + \|\eta_t \mathbf{g}_t^\ell + \eta_{t+1/2}\mathbf{g}_{t+1}^r\|_2^2 + 4\eta_{t+1/2}\eta_t G^2 \\ &\leq \|\mathbf{w}_t - \mathbf{w}^*\|_2^2 + 2\eta_t(\ell_t(\mathbf{w}^*) - \ell_t(\mathbf{w}_t)) \\ &\quad + 2\eta_{t+1/2}(r_t(\mathbf{w}^*) - r_t(\mathbf{w}_{t+1})) + 8\eta_t^2 G^2 \ . \end{aligned} \tag{26}$$

From equation (26), we finish the proof of Lemma 2.

Next, we prove the upper bound of SGFT using Lemma 2. Zinkevich's regret analysis [17] for online convex programming is effective, thus, we use this method.

From Lemma 2, when we set $\eta_t = \eta_{t+1/2}$, we obtain

$$\begin{aligned}&\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*) + r_t(\mathbf{w}_{t+1}) - r_t(\mathbf{w}^*) \\ &\quad \leq \frac{1}{2\eta_t}\left(\|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2^2\right) + 4G^2\eta_t \ . \end{aligned} \tag{27}$$

Then, we calculate the sum of equation (27) from $t = 1$ to T and derive

$$\begin{aligned}R_{\ell+r}(T) &\leq 2GD + \sum_{t=1}^{T} \frac{1}{2\eta_t}\left(\|\mathbf{w}_t - \mathbf{w}^*\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}^*\|_2^2\right) + 4G^2\sum_{t=1}^{T}\eta_t \\ &\leq 2GD + \frac{D^2}{2\eta_1} + \frac{D^2}{2}\sum_{t=2}^{T}\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) + 4G^2\sum_{t=1}^{T}\eta_t \\ &\leq 2GD + \frac{D^2}{2\eta_T} + 4G^2\sum_{t=1}^{T}\eta_t \ , \end{aligned} \tag{28}$$

from the following restriction

$$\sum_{t=1}^{T}(r_t(\mathbf{w}_t) - r_{t-1}(\mathbf{w}_t)) - r_T(\mathbf{w}_{T+1}) \leq \|\partial r_T(\mathbf{w})\|\|\mathbf{w}\|_2 \leq 2GD \ . \tag{29}$$

The second inequality holds using $\|\mathbf{w}_t - \mathbf{w}^*\|_2 \leq D$. Assuming that $\eta_t = c/\sqrt{t}$, we can prove the upper bound of regret is $O(\sqrt{T})$ from the fact that $\sum_{t=1}^{T}\eta_t \leq 2c\sqrt{T}$. Thus, we have proved Theorem 1.

A Shapley Value Approach for Influence Attribution

Panagiotis Papapetrou[1,2], Aristides Gionis[3], and Heikki Mannila[1,2]

[1] Department of Information and Computer Science, Aalto University
[2] Helsinki Institute for Information Technology (HIIT), Finland
[3] Yahoo! Research, Barcelona, Spain

Abstract. Finding who and what is "important" is an ever-occurring question. Many methods that aim at characterizing important items or influential individuals have been developed in areas such as, bibliometrics, social-network analysis, link analysis, and web search. In this paper we study the problem of attributing influence scores to individuals who accomplish tasks in a collaborative manner. We assume that individuals build small teams, in different and diverse ways, in order to accomplish atomic tasks. For each task we are given an assessment of success or importance score, and the goal is to attribute those team-wise scores to the individuals. The challenge we face is that individuals in strong coalitions are favored against individuals in weaker coalitions, so the objective is to find fair attributions that account for such biasing. We propose an iterative algorithm for solving this problem that is based on the concept of Shapley value. The proposed method is applicable to a variety of scenarios, for example, attributing influence scores to scientists who collaborate in published articles, or employees of a company who participate in projects. Our method is evaluated on two real datasets: ISI Web of Science publication data and the Internet Movie Database.

Keywords: Shapley value, influence attribution, impact factors.

1 Introduction

People have always been intrigued by characterizing "influential" ideas, books, inventions, scientists, politicians, art movements, etc. The question of finding important or influential items has attracted a lot of attention in social sciences, computer science, as well as other fields. For instance, the analysis of social networks has developed many methods to identify "important" individuals. Research in bibliometrics has provided many methods, typically based on citations, in order to assess the "impact factor" of publications, conferences, or journals. In information retrieval and web search, link-analysis methods, such as PageRank [18] and HITS [14], aim at identifying "authoritativeness" of web documents; furthermore, those link-analysis ideas have been applied to a wide variety of other scenarios. Recently, with the explosion of user-generated content a lot of emphasis has been placed on identifying influential users in blogs, micro-blogs, question-answering portals, and other social-media sites.

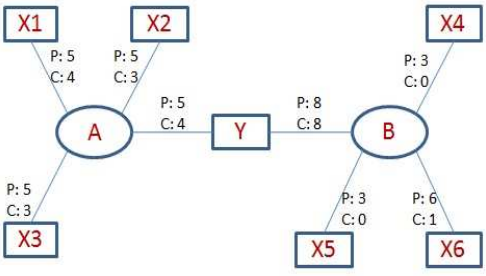
D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 549–564, 2011.

In this paper we address a novel problem in the context of characterizing who is influential. In particular, we focus on the case of attributing influence scores to a set of individuals who accomplish tasks in a collaborative manner. The setting we consider is as follows: we start with a large set of individuals; based on their interests, expertise, skills, and social dynamics, individuals form small teams in order to accomplish various tasks; we assume that for each of the accomplished tasks a value of importance (or influence) can be obtained. The problem we address is how to attribute to the individuals of each team the influence scores that we obtain for the whole team. We call this problem the *influence-attribution problem*.

Influence attribution has applications in many scenarios in which we want to estimate importance of individuals when importance evaluation scores are available at a team level. An obvious application of our setting is estimating importance scores for scientists through analysis of co-authorship information in scientific publications. For the latter problem, assigning importance scores to specific publications has received a lot of attention, and it is typically accomplished via analysis of citation graphs; however, how to distribute those scores to the individual authors of the publications is a much less studied problem. Other application scenarios include finding importance of company employees who participate in various projects, editors of wikipedia articles, bloggers in sites of collective blogging, artists in collaborative art spaces, and in general in any online collaboration community.

A simple approach for attributing influence scores to individuals is to divide the score of a task equally to all team members. However, in many cases, an equal division of importance scores may introduce inaccuracies. For instance, imagine a very talented individual who is capable of creating influential work in a number of different teams with a wide range of collaborators. In such a case, if there is evidence in the data that a large part of the influential work is due to this talented individual, it should be appropriate that she will receive more credit than her collaborators. The following example, in the context of scientific publications, demonstrates in more detail that equal division of importance score may be misleading.

Example. Consider the example shown in Figure 1. Let B be a researcher who, during a postdoctoral period, collaborated with a very distinguished researcher Y, and together they produced a total of 8 publications. Each of these publications has so far acquired 8 citations. Since the end of the postdoctoral period, B produced 12 additional publications, collaborating with 3 other researchers, but not with Y. None of these additional publications, however, received a significant amount of citations. Based on Figure 1, the total number of publications produced by A is 20, the total citation count is 70, while the H-index is 8. In the meantime, researcher A also spent a postdoctoral period collaborating with Y and produced 5 publications. Each of these publications received 4 citations. After the postdoctoral period, A collaborated with 3 other researchers producing 15 additional publications. In particular, each publication produced by the collaboration with *X1* received 4 citations and each publication produced by the



Author	Total number of papers	Total citation count	H-index
A	20	70	4
B	20	70	8

Fig. 1. An example showing that taking into account author coalitions is important. P denotes the number of papers per author and C the number of citations per paper. Thus, author $X1$ received 20 citations, authors $X2$ and $X3$ received 15 citations, authors $X4$ and $X5$ 0 citations, and author $X6$ 6 citations. Also, author Y received 84 citations. For simplicity, we have kept the citation numbers unrealistically small.

collaboration with *X2* and *X3* received 3 citations. Thus, the total number of publications of A is 20, the total citation count is 70, while the H-index is 4.

The key question now is the following: which of the two researchers is more "influential"? Taking into account the bibliometric measures, B should be favored for having a higher H-index than B. But looking a bit deeper, one may notice that B managed to obtain all the "fame" only because of collaborating with Y. So, isn't it just that Y is a very strong researcher or has a very strong team that makes B influential? In other words, if Y is dropped from the picture, then the remaining publications of B are insignificant. On the other hand, A has also collaborated with the same number of people as B, though all collaborations were fruitful. Thus, it is more likely that A is a much stronger researcher than B, and should be definitely favored.

One of course could make the following alternative conclusion: A has better selected collaborators but, in general, he/she is weak. However, in this case it is not just that A made a wise selection of collaborators, but these collaborators chose A as well. Thus, A should be strong since he/she has been chosen by strong collaborators.

Our approach. Our solution to the problem of influence attribution is inspired by the concept of *Shapley value*, which was introduced by Lloyd Shapley in 1953 [22]. The Shapley value is a game-theoretic concept devised for fair division of gains obtained by co-operation of n players, namely in a setting very similar to the one we consider in this paper. It can be shown that the division made by the Shapley value satisfies very natural fairness properties, such as efficiency, individual fairness, symmetry, and additivity [23]. However, applying the concept of Shapley value in real scenarios is highly impractical, since the theory assumes

that it is possible to assess the expected gain for every possible co-operation, that is, for all 2^N possible teams, where N is the total number of individuals.

To address the inefficiency that stems from the definition of Shapleyvalue, we propose an iterative algorithm that aims to approximate the Shapley value using only co-operations for which an estimate of the expected importance score exists. For example, in the case of co-authorships and scientific publications, the only co-operations used by the algorithm are teams of authors who have written a paper together. Even though the total number of co-operations that occur in real data is very small compared to the number of all possible co-operations, and there is no theoretical guarantee that the scores obtained by our method satisfy any fairness conditions, our approach takes into account the marginal contribution of the individuals to the teams into which they participate. Thus, we argue and experimentally demonstrate that our method produces more fair attributions than the simple baseline of equal division.

Contributions. The main contributions of this paper include:

- the formulation of the problem of finding influential individuals in a collaborative environment,
- an iterative method to solve the influence-attribution problem that is based on the Shapley value,
- an experimental evaluation on two large and commonly used datasets: ISI Web of Science and the Internet Movie Database.

The remainder of this paper is organized as follows: in Section 2 we present the related work, in Section 3 we provide the necessary background along with the problem formulation, Section 4 presents the proposed methodology, in Section 5 we present the experimental results, and finally, in Section 6 we conclude the paper and discuss directions for future work.

2 Related Work

Social-network analysis. Researchers in social-network analysis have developed many methods to measure "importance" of individuals in an implicitly- or explicitly-defined social network. In the basic model, the network is represented by a *directed* graph $G = (V, E)$, where the nodes represent individuals, and the edges model "endorsement" from one individual to another. For such a directed graph, the concept of *in-degree* is the simplest measure of importance of an individual. Refinements of the in-degree measure include the Katz-index [12] and the Hubbell index [11]. Notions of importance in social networks have been proposed also for the case of undirected graphs, most of those measures rely on various notions of *centrality*, e.g., degree centrality, closeness centrality, betweenness centrality, eigenvector centrality, etc. For an overview of those notions see the manuscript of Newman [17]. A more interesting centrality concept, more closely related with this paper, is the concept of *game-theoretic centrality* [1,9].

Bibliometrics. Research in bibliometrics studies the use of citations in order to measure the "impact" of scientific articles (e.g., journal articles, conference

proceedings) or publication venues (e.g., peer-reviewed conference proceedings, scientific journals). A large number of measures has been introduced, again starting from the simplistic citation count. The well-known *Garfield's impact factor* [7] is the average number of citations by articles published the previous two years, while the *Crown indicator* [21] is a normalized version of such a citation count. Pinski and Narin [19] and subsequently Geller [8] observe that not all citations are equally important, and using a recursive definition similar to PageRank [18,20] and HITS [14], they propose that a journal is "influential" if it is cited by other "influential" journals. The above measures are used to assign impact scores on publication "units" (e.g., articles, journals) and not to individual authors. Thus, the use of such measures is orthogonal and complementary to our work, where we aim attribute those impact scores for articles to individual authors. On the other hand, measures such as the *H-index* [10], and the *G-index* [6] (defined in Section 5) have been proposed for assigning impact scores to individual authors, and thus, are directly comparable to our proposed measure. However, both H-index and G-index do not attempt to address the issue of coalition bias as our method does. Overall, any bibliometrics impact index needs to be used with extreme care. As such our method is intended to complement and enrich existing analytics toolboxes, rather than substitute well-established methods.

Web search rankings. A prominent application domain of importance measures is in the area of web search and hypertext ranking. The goal is to assign importance scores to web documents in order to assist users locate the most relevant results for their searches. Not surprisingly many of the importance measures discussed above can also by used in the case of web search ranking, however, the two most well-known techniques are PageRank [18] and HITS [14]. Many variants of those methods have been proposed, as well as adaptations of those basic methods for different objectives; a thorough survey is beyond the scope of this paper.

Information diffusion and viral marketing. A different setting for studying influential individuals in social networks is through the concepts of *information diffusion* and *viral marketing*. This setting is described by a dynamic process. It assumes a model of information spread in the social network, and influential individuals are those who can act as good initiators of the information spread. Building on the seminal work of Domingos and Richardson [5] and Kempe et al. [13], a large number of papers has studied variants of the problem and proposed different solutions, including game-theoretic and Shapley value-based solutions such as the work of Narayanam and Narahari [16]. However, the overall setting of information-diffusion processes is very different from the problem studied in this paper.

Coalitional games and Shapley value. Studying coalitional games is a major area of research in game theory. In contrast to competitive games, to which the standard solution concept is the Nash equilibrium, in coalitional games the players cooperate and receive collective payoff. The Shapley value [22] is the

classic solution concept for distributing the payoff "fairly" among the members of the coalition. The model of coalitional game theory provides a rich framework for studying a variety of solution concepts and their properties [2,3,4]. Finding the allocations described by these concepts is in general computationally intractable, hence, much of the research in this area focuses on a variety of restricted domains in which one can hope to find approximate solutions. An overview of the theory of coalitional games is beyond the scope of the paper. However, a key difference from our work is that in this line of research approximate solutions are searched by allowing to sample random coalitions. Instead, we follow a more pragmatic approach where only a small number of coalitions is given as input, and we seek solutions by probing only those coalitions.

3 Problem Setting

We consider a set $\mathcal{V} = \{V_1, \ldots, V_n\}$ of n individuals and a set $\mathcal{T} = \{T_1, \ldots, T_m\}$ of m tasks. Each individual $V_i \in \mathcal{V}$ has participated in a subset of tasks of $\mathcal{T}$, which we denote by $\mathcal{T}_{V_i}$. Moreover, each task T_j is given an *impact score* $I_j = f(T_j)$; the higher the impact score the more "valuable" the task. Examples for function $f(\cdot)$ will be provided in Section 3.1.

Based on the participation of individuals to tasks, and the impact score of each task, our goal is to estimate the *influence score* ϕ_i of each individual $V_i \in \mathcal{V}$. Here, we assume that influence scores are non-negative numbers such that $\phi_i \geq \phi_j$ implies that V_i is at least as influential as V_j.

Hence, the *influence-attribution problem* can be formalized as follows:

Problem 1 ***(Influence-attribution)*** **.** *Given a set of individuals* $\mathcal{V} = \{V_1, \ldots, V_n\}$*, a set of tasks* $\mathcal{T} = \{T_1, \ldots, T_m\}$*, and a set of impact scores* $\mathcal{I} = \{I_1, \ldots, I_m\}$*, with* $I_j = f(T_j)$*, compute the set of influence scores* $\phi = \{\phi_1, \ldots, \phi_n\}$*, where* ϕ_i *is the influence score of individual* $V_i \in \mathcal{V}$*.*

3.1 Example: Author-Publication Instantiation

Let us now instantiate the above problem setting by considering a task to be a scientific publication and an individual to be a scientist. We call this the *author-publication* instantiation. In such setting, there may exist citations among different publications, so we define one additional concept.

Definition 1 ***(Incoming Citations).*** *Given a set of publications* $\mathcal{T}$*, for each publication* $T_j \in \mathcal{T}$ *we define the set of* incoming citations *to* T_j *as follows:*

$$\mathcal{C}^{in}_{T_j} = \{T_k \mid T_k \in \mathcal{T} \text{ and } T_k \text{ cites } T_j\}. \tag{1}$$

As mentioned above, $I_j = f(T_j)$ is the impact score of publication $T_j \in \mathcal{T}$. For the author-publication instantiation, we consider two options for $f(\cdot)$:

- `CC`: the citation count of each publication corresponds to the total number of citations received by the publication.

$$f(T_j) = |\mathcal{C}^{in}_{T_j}|, \quad \text{for each } T_j \in \mathcal{T}. \tag{2}$$

– PR: the PageRank score of each publication, which is computed by applying the PageRank algorithm on the citation network.

The definition of CC is simple enough and rates each incoming citation equally. The PR score is computed by the well-known algorithm for ranking web documents [18], which is based on estimating the stationary distribution of a random walk in the citation graph.

4 Methods

We present two methods for solving Problem 1. The first one is a straightforward uniform assignment of impact scores to the corresponding individuals, whereas the second one exploits the Shapley value in order to take into account coalitions of individuals and the impact score in these coalitions.

4.1 Naïve Approach

The first approach to solve Problem 1 is to assign individuals with the mean impact score of their tasks.

Definition 2 ***(Mean Impact).*** *Given an individual V_i, the set of tasks $\mathcal{T}_{V_i}$ assigned to V_i, and their corresponding impact scores $\mathcal{I} = \{I_1, \ldots, I_{|\mathcal{T}_{V_i}|}\}$, the mean impact of V_i is defined by*

$$\phi_i = \frac{1}{|\mathcal{T}_{V_i}|} \sum_{T_j \in \mathcal{T}_{V_i}} I_j. \tag{3}$$

For the remainder of the paper, this method will be denoted as Naïve.

4.2 The Shapley Value Approach

The second method is based on the concept of Shapley value [15], which is a way to divide goods gained by cooperation among many individuals. Specifically, consider an underlying set $\mathcal{V}$ and assume that for all possible subsets $\mathcal{S} \subseteq \mathcal{V}$, we know the value of $v(\mathcal{S})$, called *gain function*, which expresses the gain achieved by the cooperation of the individuals in $\mathcal{S}$.

Definition 3 ***(Share Allocation).*** *Given a gain function $v(\cdot)$, the* share allocation *$\phi_i(v)$ to individual V_i is defined as:*

$$\phi_i(v) = \sum_{\mathcal{S} \subseteq \mathcal{V}} \frac{|\mathcal{S}|!(|\mathcal{V}| - |\mathcal{S}| - 1)!}{|\mathcal{V}|!} (v(\mathcal{S} \cup \{V_i\}) - v(\mathcal{S})). \tag{4}$$

The intuition is to consider the marginal utility that V_i brings in set $\mathcal{S}$, averaged over all possible sets. Notice that the average is weighted. In fact, the process can be seen not as averaging over sets, but averaging over all possible permutations – assume that coalitions are generated by adding one individual at a time – so the weight of a set is the number of permutations that produce each coalition.

The definition of Shapley value is very attractive, as it can be shown theoretically that the resulting attribution satisfies natural fairness properties [23]. However, a direct application of Definition (4) in our setting is not possible. Not only it assumes an averaging over exponentially many sets, but also it is not possible to probe arbitrary sets $\mathcal{S}$ and obtain $v(\mathcal{S})$; for example, we do not have available the impact score of papers for every possible subset of authors! To address this difficulty we introduce the following ideas:

- First, we do not compute marginal gains by averaging over all possible coalitions, but only over coalitions for which we have available impact scores. The details are given in the next sections where we describe our iterative algorithm.
- Second, in order to average in a marginal contribution $v(\mathcal{S} \cup \{V_i\}) - v(\mathcal{S})$ we need to have available both values $v(\mathcal{S} \cup \{V_i\})$ and $v(\mathcal{S})$. However, in many cases we have available only one of the two. Simply ignoring those cases would lead to very sparse data. Therefore, we choose to take into account all cases for which $v(\mathcal{S} \cup \{V_i\})$ is available. If for those cases $v(\mathcal{S})$ is not available, we propose to approximate it by composing it from its subsets.

Now, we discuss how to define the values $v(\mathcal{S})$ and reasonable approximations of $v(\mathcal{S})$ when those values are not directly available. The first issue is that in some cases the same coalition has accomplished many different tasks, and each task has different impact factors; for example, a given set of authors may have many different publications. We then define the value of the coalition as the average of the impact scores of all the tasks into which the coalition has participated.

Definition 4 ***(Shared Impact Factor).*** *Consider a set of individuals $\mathcal{S}$ and let $\mathcal{T}_\mathcal{S}$ be the set of their common tasks. Let I_j be the impact factor of each common task $T_j \in \mathcal{T}_\mathcal{S}$. Then, the* shared impact factor *of $\mathcal{S}$ can be defined as:*

$$v(\mathcal{S}) = \frac{1}{|\mathcal{T}_\mathcal{S}|} \sum_{j=1}^{|\mathcal{T}_\mathcal{S}|} I_j. \tag{5}$$

The next issue that we need to address is that, during the execution of our iterative algorithm, we need to use the value of $v(\mathcal{S})$, for cases that we do not have any information about coalition $\mathcal{S}$ (e.g., there is no paper with that exact set of authors). Thus, we propose to approximate the value of $v(\mathcal{S})$ by taking into account subsets $\mathcal{S}' \subseteq \mathcal{S}$ for which we have information about the impact of their coalition. We suggest to average over all subsets of $\mathcal{S}$ for which we have information, and also include a term to capture the contribution of the individuals who do not form any coalition in subsets of $\mathcal{S}$.

Definition 5 ***(Approximated Shared Impact Factor).*** *Consider a set of individuals $\mathcal{S}$ and let $\mathcal{S}^c = \bigcup_i \mathcal{S}_i^c$ be the set of all subsets $\mathcal{S}_i^c$ of $\mathcal{S}$ where all individuals have at least one common task. Then, the* approximated shared impact factor *of the individuals in $\mathcal{S}$ is:*

$$v'(\mathcal{S}) = \frac{1}{|\mathcal{S}^c| + 1} \left(\sum_{i=1}^{|\mathcal{S}^c|} v(\mathcal{S}_i^c) + \bar{v}(\mathcal{S} \setminus \mathcal{S}_i^c) \right), \tag{6}$$

where $v(\mathcal{S}_i^c)$ can be computed using Equation (5) and $\bar{v}(\mathcal{S} \setminus \mathcal{S}_i^c)$ is an estimation of the impact of individuals who do not form any coalition in subsets of $\mathcal{S}$.

To complete the definition of function $v'(\cdot)$ we need to define function $\bar{v}(\cdot)$, which we use when individuals do not participate in any common coalition. We do this by using a recursive definition. We choose to compute $\bar{v}(\mathcal{S})$ using the influence factors ϕ_i of the individuals who form coalition $\mathcal{S}$. Assuming a monotonic behavior, that is, assuming that teams are at least as good as the best individual who composes the team, we define $\bar{v}(\cdot)$ using the *maximum* operator.

Definition 6 ***(Approximated Gain Function).*** *The* approximated gain function $\bar{v}(\mathcal{S})$ *is defined as follows:*

$$\bar{v}(\mathcal{S}) = \max_{V_i \in \mathcal{S}} \phi_i(v). \tag{7}$$

We note that Definition (6) is recursive since our goal is to compute the scores ϕ_i. Therefore, our definition suggests an iterative algorithm, described in more detail in the next section: the algorithm starts with initial estimates of the scores ϕ_i, and then it iteratively adjusts the estimate of those scores until convergence.

4.3 The Iterative Algorithm

We propose an iterative algorithm to compute the influence score ϕ_i of each individual $V_i \in \mathcal{V}$. At each iteration t, the value of the influence score is denoted as ϕ_i^t. At first, the influence scores of each individual are initialized to the average impact factor his/her tasks. Then, at each iteration, the Shapley value is computed using Definition (4). The key idea behind our algorithm is to compute the Shapley value averaging over a small set of coalitions, for which the impact factor is available. Whenever the algorithm needs to probe a coalition whose impact factor is not available, then the approximated shared impact factor is used (Definition 5).

During the main loop of the algorithm, for each available task $T_j \in \mathcal{T}$ and each individual V_i of T_j, we update the influence score $\phi_i^{t+1}(v')$ by adding the difference of the gain function of sets $\mathcal{V}_{T_j}$ and $\mathcal{V}_{T_j} \setminus V_i$. The same weighting factor is used, as in Equation (4), that is,

$$\phi_i^{t+1}(v') = \sum_{\mathcal{V}_{T_j} | V_i \in T_j} \frac{|\mathcal{V}_{T_j}|!(|\mathcal{V}| - |\mathcal{V}_{T_j}| - 1)!}{|\mathcal{V}|!}(v'(\mathcal{V}_{T_j}) - v'(\mathcal{V}_{T_j} \setminus V_i)). \tag{8}$$

This procedure continues until convergence, which is defined by the following criterion.

$$\frac{\sum_{i=1}^{|\mathcal{V}|} |\phi_i^t - \phi_i^{t-1}|}{\sum_{i=1}^{|\mathcal{V}|} \phi_i^{t-1}} \leq \epsilon \in (0, 1). \tag{9}$$

The pseudocode of this method is given in Algorithm 1. For the remainder of this paper, this method will be denoted as `Shapley`.

Algorithm 1. The `Shapley` Algorithm

Input: a set of individuals $\mathcal{V}$, a set of tasks $\mathcal{T}$, and the corresponding set of impact scores $\mathcal{I}$.
Output: the influence score ϕ_i of each individual $V_i \in \mathcal{V}$
// Initialization: $\forall T_i, i = 1, \ldots, m$ assigned to individual V_i:
for $j = 1 : |\mathcal{V}|$ **do**
  $\phi_i^0 = \sum_{i=j}^{m} I_j$
end for
while convergence **do**
  Initialize $\phi_i^{t+1}(v') = 0$
  for $T_j \in \mathcal{T}$ **do**
    for $V_i \in \mathcal{V}_{T_j}$ such that V_i is assigned with task T_j **do**
      $\phi_i^{t+1}(v') = \phi_i^{t+1}(v') + \frac{|\mathcal{V}_{T_j}|!(|\mathcal{V}|-|\mathcal{V}_{T_j}|-1)!}{|\mathcal{V}|!}(v'(\mathcal{V}_{T_j}) - v'(\mathcal{V}_{T_j} \setminus V_i))$
    end for
  end for
end while

4.4 Enforcing Monotonicity of the Gain Function

In the theory of coalition games, the gain function is assumed to be monotone and non-negative. Here monotonicity means that if $\mathcal{S}_1 \subseteq \mathcal{S}_2$ then $v(\mathcal{S}_1) \leq v(\mathcal{S}_2)$. Those assumptions are also desirable in our setting. Thus, we enforce those conditions by first computing all payoffs—$v(\mathcal{S})$—that are available for each coalition $\mathcal{S}$. Then, we identify all pairs of sets $\mathcal{S}_1$ and $\mathcal{S}_2$, such that $\mathcal{S}_1 \subseteq \mathcal{S}_2$ and $v(\mathcal{S}_1) > v(\mathcal{S}_2)$. For each such pair we increase $v(\mathcal{S}_2)$ by setting $v(\mathcal{S}_1) = v(\mathcal{S}_2)$. This is repeated until all violations in these pairs are eliminated. In other words, we define the partial order of all coalition sets, and then we guarantee that while paths of set inclusion are followed, the payoffs defined by the gain function do not increase. Once the monotonicity property is satisfied, the computations performed by our algorithm ensure that the gain function is also non-negative.

5 Experiments

5.1 Setup

We evaluated the performance of the proposed method on two real datasets: one bibliographic dataset and one movie dataset.

ISI Web of Science. The first dataset is part of the Thomson Reuters ISI Web of Science data. ISI covers mainly journal publications. We sampled data related to our institutions published within years 2003 and 2009. Our dataset contained data about 1212 authors.

We used two very common bibliometric indicators as the baseline:

- **H-Index:** a scientist's H-index is h, if h of his/her publications have at least h citations and the rest of his/her publications have at most h citations each.

– **G-Index:** a scientist's G-index is g, if g of his/her highly cited publications received, together, at least g^2 citations.

We investigated both types of impact scores discussed earlier (Section 3.1) for the author-publication scenario.

Internet Movie DataBase. The second dataset is part of the Internet Movie Database (IMDB) data.[1] We sampled a total of 2 000 male actors and 4 560 movies. We restricted the movie genre type to "comedy" or "action", and did not include "TV series". For each actor we considered only the movies where his credit position was among the top 3. Each movie T_j was assigned with an impact score I_j defined to be $I_j = R_j A_j$, where R_j is the average rating received by movie T_j and A_j is the number of people who evaluated this movie.

To evaluate the performance of our methods we used the following measure:

Definition 7 ***(rank of individual).*** *Given a set of individuals* $\mathcal{V} = \{V_1, \ldots, V_n\}$ *and their influence scores* $\phi = \{\phi_1, \ldots, \phi_n\}$*, the rank of individual* V_i *is* r*, if*

$$|V_j | \phi_j \geq \phi_i, j \in 1, \ldots, n| = r. \tag{10}$$

In other words, the rank of an individual V_i measures the number of individuals who are at least as influential as V_i.

5.2 Experimental Results

ISI Web of Science. We compared the performance of `Naïve` and `Shapley` on the ISI Web of Science data sample described in Section 5.1. Each of the two methods computed an impact score per author. We then compared the author ranks of the proposed methods with the ranks obtained when computing the `H-index` and `G-index`. In Figure 2, we can see the performance of the proposed methods with respect to author ranks, compared to `H-index`. It is evident that there is a high correlation between `H-index` and `Naïve-CC` (the latter denotes the version of `Naïve` that uses the average citation score for defining function $f(\cdot)$). This correlation becomes much weaker in the case of `Naïve-PR`, where PageRank is used, implying that considering higher-order citations (as done by PageRank) can cause a significant change in the ranking of publications and, consequently, in the ranking of authors. In the case of `Shapley`, we see that as the `H-index` rank increases, the variance in the rank obtained by `Shapley` also increases. The results with respect to `G-index` are very similar but omitted due to space limitations.

Next, we compared the performance of `Naïve` and `Shapley` with respect to author ranks. Again, we investigated both cases of impact scores for the publications: citation count and PageRank. In Figure 3 we see a comparison of the ranks produced by the four methods. When comparing `Naïve` and `Shapley` using citation count the two methods deviate as ranks increase. The same behavior

[1] `http://www.imdb.com/`

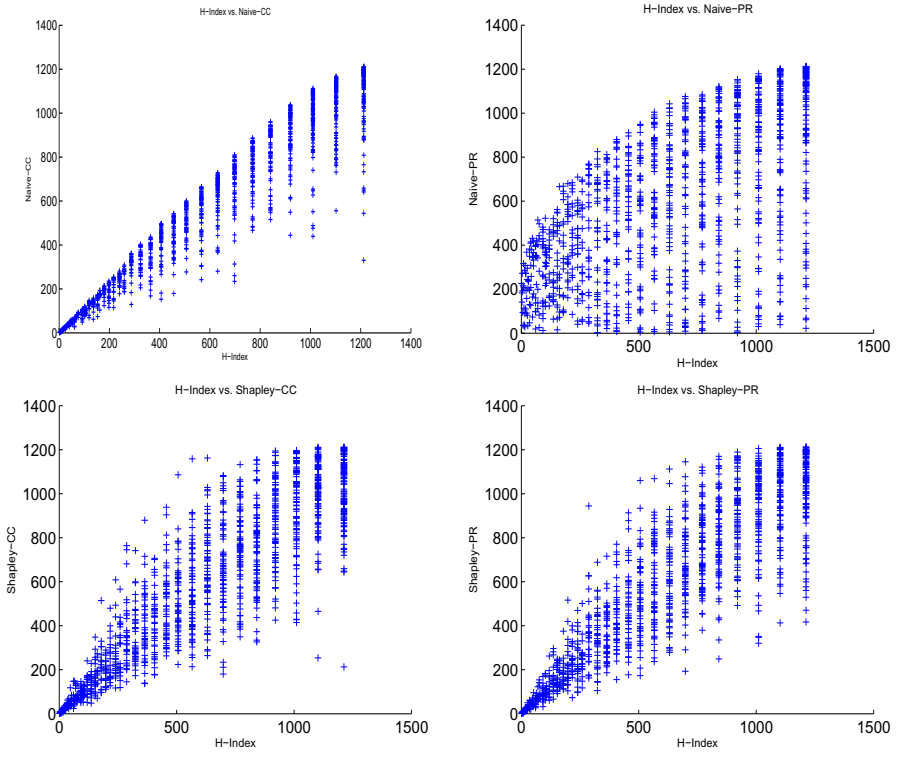


Fig. 2. Performance of `Naïve` and `Shapley` compared to H-index, with respect to author ranks, for the ISI Web of Science data. Note that the values in the x and y-axis of the figures correspond to ranks.

is noticed when comparing `Shapley` using citation count vs. using PageRank. This effect is more intense for `Naïve`. In the latter case we note that `Naïve-CC` generally increases the ranking of the publications, compared to `Naïve-PR`. The results with respect to `G-index` are very similar.

We compared the ranks of the top-10 authors retrieved by `Shapley-CC`, Table 1(left) and `Shapley-PR`, Table 1(right), to the ranks produced by the other methods. To preserve the anonymity of the scientists we do not show their names in the table. An interesting observation is that the deviation in the ranks between the methods is much higher for `Shapley-PR` than for `Shapley-CC`.

Internet Movie Database. The performance of `Naïve` and `Shapley` was also evaluated on the IMDB data sample described in Section 5.1. In this case there are no baseline indicators, such as `H-index`, hence we only compared `Naïve` and `Shapley`. In Table 2(left) we see the names of the top-10 actors given by `Shapley` and the corresponding ranks given by `Naïve`. It appears that actors who participated in the casting of very highly rated movies are favored against those who also participated in low rated movies. It is interesting to see that `Shapley`

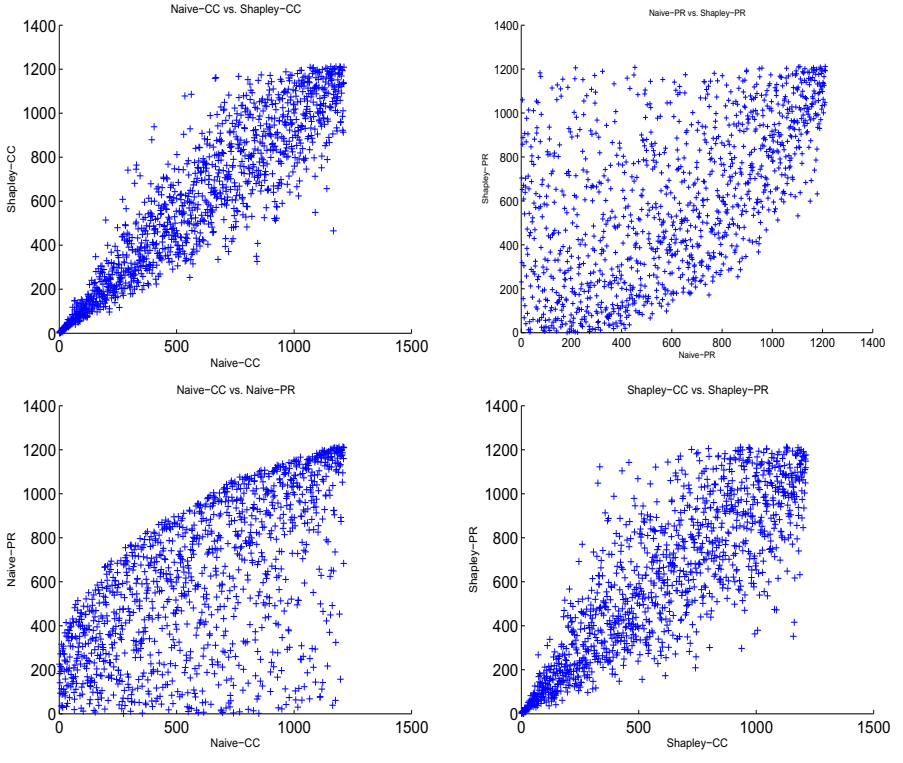


Fig. 3. Comparison of the `Naïve` and `Shapley` methods with respect to author ranks, for the ISI Web of Science data. Note that the values in the x and y-axis of the figures correspond to ranks.

managed to detect "big" names in *comedy* and *action*. In Table 2(right) we show the top-10 actors discovered by `Naïve` and their corresponding ranks by `Shapley`. Deviations between the two methods suggest that actors may manage to achieve a high mean movie rating (`Naïve`) but still there are several poorly rated movies with highly rated co-actors that cause a drop in their Shapley score.

Four more examples of deviating ranks are shown in Table 3. We can see that several famous actors show a high deviation in their ranking. For example, "Adam Sandler" is ranked 59^{th} by `Shapley` while `Naïve` ranks him 4^{th}. Based on the definition of the Shapley value, this implies that he may have highly ranked movies, but he also has movies where he is co-acting with other "strong" actors; these movies have a higher ranking than the ones where he is co-acting with "weaker" actors. Similar conclusions can be made for "Jim Carrey". On the one hand, he has a significant amount of highly rated movies, but also some low rated movies (which drops the score attributed to him by `Naïve`); on the other hand, in almost all of his movies he has been the main and "strongest" actor (which results in `Shapley` attributing him a high score).

Table 1. List of top-10 authors given by `Shapley-CC` (left) and by `Shapley-PR` (right). For `Shapley-CC` we used the citation count to assign impact scores to the publications, while for `Shapley-PR` we used the PageRank score. All values in the table correspond to ranks.

`Shapley-CC`	`Naïve-CC`	H-index	G-index	`Shapley-PR`	`Naïve-PR`	H-index	G-index
1	1	1	2	1	183	2	1
2	3	3	4	2	225	1	2
3	6	8	11	3	35	8	9
4	5	4	4	4	215	8	11
5	4	6	5	5	192	5	7
6	12	25	22	6	272	4	4
7	10	9	11	7	94	46	25
8	2	2	1	8	141	23	14
9	8	8	9	9	208	11	13
10	15	15	14	10	114	11	7

Table 2. List of actors ranked as top-10 by `Shapley` (left) and by `Naïve` (right), and their corresponding ranks obtained by the other methods. All values in the table correspond to ranks.

Actor Name	`Shapley`	`Naïve`	Actor Name	`Naïve`	`Shapley`
Robert De Niro	1	3	Peter Sellers	1	14
Al Pacino	2	8	Jack Nicholson	2	11
Brad Pitt	3	15	Robert De Niro	3	1
Bruce Willis	4	7	Adam Sandler	4	59
Arnold Schwarzenegger	5	24	Daniel Day-Lewis	5	36
Will Smith	6	13	Chris Farley	6	20
Eddie Murphy	7	10	Bruce Willis	7	4
Robin Williams	8	9	Al Pacino	8	2
Morgan Freeman	9	17	Robin Williams	9	8
Ben Stiller	10	29	Eddie Murphy	10	7

Table 3. A list of 4 examples of actors with high deviation between `Shapley` and `Naïve`. All values in the table correspond to ranks.

Actor Name	`Shapley`	`Naïve`	# of Movies in IMDB	Average Movie Rating
Jim Carrey	11	79	34	5.2
Sylvester Stallone	12	41	46	6.4
Daniel Day-Lewis	36	5	27	7.1
Adam Sandler	59	4	39	5.4

6 Conclusions

We addressed the problem of attributing influence to a set of individuals who participate in a set of tasks. The method we proposed employs the game-theoretic concept of Shapley value. We showed that the proposed methodology can be

applied in many real scenarios, for example, in the author-publication scenario for assigning influence scores to scientists who collaborate in published articles, or in the movie-actor scenario for evaluating actors who collaborate in different movies. We evaluated our method on two real datasets, and showed that it highly differs with respect to ranking when compared to a naïve approach of equal division of influence.

In the author-publication scenario, we illustrated by an example (Figure 1), and also experimentally, that existing bibliometrics indicators, such as `H-index` and `G-index`, do not take into account the average strength of author coalitions and thus may favor authors with a few highly cited publications that resulted from a single (or very few) strong collaborators. Similar conclusions were drawn from the movie data, where very famous actors were given a lower ranking by `Shapley` than by `Naïve`. These actors were disfavored since their "fame" was caused only by a few specific movies and co-actors.

Directions for future work include the investigation of other domains such as user-blogs and social-media sites. Moreover, one could study how to use additional information regarding the involved tasks or individuals. For example, in the actor-movie scenario, several movie features are available as well as information about the actors. This additional information may or should affect the share allocation function. Finally, a challenging task is to further evaluate the quality of the rankings obtained, possibly by performing user studies.

Acknowledgements. This work was supported in part by the Academy of Finland ALGODAN Centre of Excellence. Additionally, this work was partially supported by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN20101037, "Social Media"[2].

References

1. Aadithya, K.V., Ravindran, B., Michalak, T.P., Jennings, N.R.: Efficient computation of the shapley value for centrality in networks. In: Saberi, A. (ed.) WINE 2010. LNCS, vol. 6484, pp. 1–13. Springer, Heidelberg (2010)
2. Aziz, H.: Algorithmic and complexity aspects of simple coalitional games. PhD thesis, University of Warwick (2009)
3. Deegan, J., Packel, E.W.: A new index of power for simple n-person games. International Journal of Game Theory 7(2) (1978)
4. Deng, X., Fang, Q.: Algorithmic cooperative game theory. Pareto Optimality, Game Theory and Equilibria 17(1), 159–185 (2008)
5. Domingos, P., Richardson, M.: Mining the network value of customers. In: KDD, pp. 57–66 (2001)
6. Egghe, L.: Theory and practise of the G-index. Scientometrics 69 (2006)
7. Garfield, E.: Citation analysis as a tool in journal evaluation. Science 178(4060), 471–479 (1972)
8. Geller, N.: COn the citation influence methodology of Pinski and Narin. Information Processing & Management 14, 93–95 (1978)

[2] www.cenitsocialmedia.es

9. Gomez, D., Gonzalez-Aranguena, E., Manuel, C., Owen, G., del Pozo, M., Tejada, J.: Centrality and power in social networks: a game theoretic approach. Mathematical Social Sciences 46(1), 27–54 (2003)
10. Hirsch, J.E.: An index to quantify an individual's scientific research output. Proceedings of the National Academy of Sciences of the United States of America 102(46), 16569–16572 (2005)
11. Hubbell, C.H.: An input-output approach to clique identification. Sociometry 28(4), 377–399 (1965)
12. Katz, L.: A new status index derived from sociometric analysis. Psychometrika 18, 39–43 (1953)
13. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: KDD, pp. 137–146 (2003)
14. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
15. MasColell, A., Whinston, M., Green, J.R.: Microeconomic Theory. Oxford University Press, Oxford (1995)
16. Narayanam, R., Narahari, Y.: A shapley value-based approach to discover influential nodes in social networks. IEEE Transactions on Automation Science and Engineering 8(1) (2011)
17. Newman, M.E.J.: The mathematics of networks. The New Palgrave Encyclopedia of Economics (2007)
18. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web (1999)
19. Pinski, G., Narin, F.: Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics. Information Processing & Management 12, 297–312 (1976)
20. Radicchi, F., Fortunato, S., Markines, B., Vespignani, A.: Diffusion of scientific credits and the ranking of scientists. Phys. Rev. E 80(5), 056103 (2009)
21. Rehn, C., Kronman, U., Wadskog, D.: Bibliometric indicators – definitions and usage at Karolinska Institutet. Technical report (2007)
22. Shapley, L.S.: A value for n-person games. Annals of Mathematical Studies 28, 307–317 (1953)
23. Winter, E.: The Shapley value. In: Handbook of Game Theory with Economic Applications, vol. 3 (2002)

Fast Approximate Text Document Clustering Using Compressive Sampling

Laurence A.F. Park

School of Computing and Mathematics,
University of Western Sydney, Australia
lapark@scm.uws.edu.au
http://www.scm.uws.edu.au/~lapark

Abstract. Document clustering involves repetitive scanning of a document set, therefore as the size of the set increases, the time required for the clustering task increases and may even become impossible due to computational constraints. Compressive sampling is a feature sampling technique that allows us to perfectly reconstruct a vector from a small number of samples, provided that the vector is sparse in some known domain. In this article, we apply the theory behind compressive sampling to the document clustering problem using k-means clustering. We provide a method of computing high accuracy clusters in a fraction of the time it would have taken by directly clustering the documents. This is performed by using the Discrete Fourier Transform and the Discrete Cosine Transform. We provide empirical results showing that compressive sampling provides a 14 times increase in speed with little reduction in accuracy on 7,095 documents, and we also provide a very accurate clustering of a 231,219 document set, providing 20 times increase in speed when compared to performing k-means clustering on the document set. This shows that compressive clustering is a very useful tool that can be used to quickly compute approximate clusters.

1 Introduction

Clustering computational complexity is dependent on the number of objects in the set and the number of features of each object. Therefore, as the number of features grows, the feasibility of applying a clustering algorithm reduces. To perform clustering, the data must be repeatedly scanned while the clusters are refined therefore it is also important that we have enough memory for the computation to avoid lengthy disk accesses.

As technology advances, the size of data to be processed also increases. Text document databases are growing at a rapid rate, therefore, it is crucial that we derive document clustering algorithms that are able to process and cluster these large data sets.

Compressive sampling [4,5,3,7,1] is a new concept in Information theory that states that we are able to perfectly reconstruct a vector from only a few samples, when using an appropriate incoherent sampling scheme.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 565–580, 2011.

In this article, we introduce compressive clustering; a method of clustering using incoherent samples of the features, that provides a close approximation to the clusters that would have been found on the unsampled data. We show that we are able to apply compressive clustering to very high dimensional spaces and obtain very accurate cluster estimates in a fraction of the time.

We make the following contributions:

- We provide a generalised algorithm for compressive clustering (Section 3),
- A radial k-means algorithm for complex vector spaces (Section 3.2), and
- Use of compressive clustering for document clustering using radial k-means clustering with discrete Fourier and discrete Cosine sampling (Section 3.3).

The article will proceed as follows: Section 2 provides a brief introduction to compressive sampling, section 3 introduces our compressive clustering algorithm and how it can be applied to document clustering using radial k-means. Section 4 provides experimental results showing the performance of compressive clustering for document clustering, and finally section 5 presents the use of compressive clustering on a large document set.

2 Coherence and Random Projections

Compressive sampling is a new sampling technique that has gained popularity in the image processing domain. It is an generalisation of the Nyquist sampling theorem that is not restricted to band-limited signals. In this section we will examine Nyquist's theorem and Compressive sampling.

2.1 Sampling Cyclic Signals

The Nyquist-Shannon sampling theory is at the base of Information Theory. The theory states that a signal containing no frequencies greater than b hertz, can be perfectly reconstructed by sampling at a rate of at least $2b$ hertz.

Figure 1 shows a simple example of a 2 hertz signal in the form of a sinusoidal wave. The Fourier transform of this signal would contain one non-zero frequency component, therefore when sampling the signal, we must ensure that we capture this single non-zero frequency value. If we can do this, and assume that the remaining frequency values are zero, then we are able to perfectly reconstruct the sinusoidal wave. The Nyquist-Shannon theorem tells us that we are able to capture this nonzero frequency component by sampling at at least two times the frequency, which in this case is a rate of 4 hertz.

2.2 Sampling Sparse Signals

The Nyquist-Shannon sampling theorem allows us to capture all frequency components up to half of our sampling rate. This allows us to efficiently encode low frequency signals, but what if there were only a few non-zero frequency components that were scattered across the frequency domain. By using the Nyquist-Shannon sampling theorem, we would have to sample at a high rate in order to

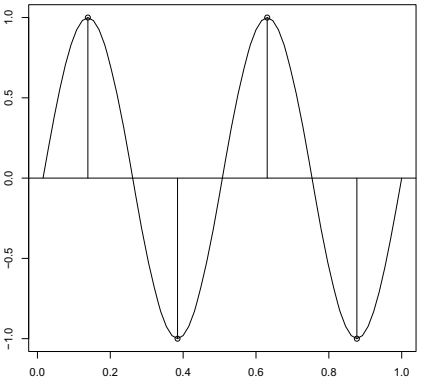


Fig. 1. A 2 hertz signal, being sampled at a rate of four hertz. According to the Nyquist-Shannon sampling theorem, we can perfectly reconstruct the signal using these four samples.

capture only a few non-zero frequency components. Or what if the frequency domain of the signal was dense (many non-zero components), but there was some other domain in which the signal has a sparse representation?

A recent advancement in Information Theory is that of Compressive Sampling [4,5,3,7,1]. Compressive sampling theory states that given a signal that is shown to be sparse in a known domain, we are able to take a small number of random samples of the signal and successfully reconstruct the signal to its original state. Stated more formally, we have:

$$\boldsymbol{x} = \min \|\boldsymbol{z}\|_1, \text{s.t. } \boldsymbol{\xi} = \Phi\Psi\boldsymbol{z} \tag{1}$$

where $\boldsymbol{y} = \Psi\boldsymbol{x}$ is the original signal and $\boldsymbol{x}$ is sparse, Φ is the sampling function, $\boldsymbol{\xi}$ is the set of samples, and $\|\cdot\|_1$ is the l_1 norm. To obtain perfect reconstruction, we must ensure that $\boldsymbol{x}$ is S-sparse (implying that $\boldsymbol{x}$ has at most S non-zero values), and that the transforms Φ and Ψ are incoherent.

Before we proceed lets examine the case where the Fourier transform of a signal is sparse, but the non-zero components are spread evenly across the whole spectrum. We have said that the Nyquist-Shannon sampling theorem would require us to sample at a high rate, due to some of the frequency components being of high frequency. By using Compressive Sampling, we know that the Fourier Transform of the signal is sparse, therefore we let Ψ be the Fourier transform and $\boldsymbol{x}$ be the transformed signal. We can construct a sampling function Φ by randomly sampling rows of the identity matrix. This implies that if we take K random samples of our signal $\Phi\boldsymbol{x}$, we are able perfectly reconstruct it given that K is large enough.

It has been shown that when using the Fourier transform as Ψ and a sampled identity matrix as Φ, we obtain the relationship for perfect reconstruction:

$$K \geq CS \log N$$

where K is the number of samples, N is the length of the original signal and C is a constant.

Therefore, given that $S = 10$, $N = 1000$, then the number of samples K required for perfect reconstruction is proportional to 70. Note that if not enough samples are taken, the resulting reconstructed sparse vector will be similar to a thresholded version of $\boldsymbol{x}$ due to the use of the l_1 norm in the minimisation.

For the example above, we chose a transform and sampling matrix that were maximally incoherent. Coherence is a measure of basis similarity that computes the smallest angle between the basis vectors from the two basis sets. The coherence of Φ and Ψ is given as:

$$\mu(\Phi, \Psi) = \sqrt{N} \max_{1 \leq i,j \leq N} |\langle \boldsymbol{\phi_i}, \boldsymbol{\psi_j} \rangle|$$

where $\boldsymbol{\phi_i}$ and $\boldsymbol{\psi_j}$ are basis vectors in the transformation Φ and Ψ, and $\mu(\Phi, \Psi) \in [1, \sqrt{N}]$. Therefore given the Fourier transform and identity matrix as Ψ and Φ, we obtain $\mu(\Phi, \Psi) = 1$, being maximally incoherent. For other choices of Ψ and Φ, where the coherence is greater than one, we have the generalised relationship:

$$K \geq C\mu^2(\Phi, \Psi) S \log N$$

Therefore, we can see that the choice of the transformation and sampling matrices are crucial in reducing the number of samples required for perfect reconstruction. The more incoherent the two basis sets are, the more they spread over each other. For example, the Fourier and identity basis are maximally incoherent since each Fourier coefficient is a combination of all identity basis vectors.

3 Compressive Clustering

Now that we have an understanding of compressive sampling, we will introduce the use of compressive sampling for clustering.

We have stated that clustering algorithms require repetitive access to the data while clustering, therefore as the size of the data grows, so to does the time and storage required to compute the clusters. This implies that there is a limit as to the size of data we can cluster, that is dependent on the current computation and storage available.

Compressive sampling has provided us with the concept of incoherent sampling. By using incoherent samples, we are able to capture enough information to perfectly reconstruct the vector from the samples, given knowledge of the sampling scheme. We can apply this theory to obtain a low dimensional feature space in which we can perform clustering. Using the small dimensional representation of the clusters, we are able to reconstruct the vectors in the original high dimensional space.

Our compressive clustering algorithm is as follows: Given a data set $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M\}$, a transform Ψ, such that $\boldsymbol{y}_i = \Psi \boldsymbol{x}_i$ where each $\boldsymbol{x}_i$ are sparse, and a sampling function Φ that is incoherent to Ψ:

1. Sample the features of the vector space using the sampling function to obtain $\boldsymbol{\xi}_i$, where $\boldsymbol{\xi}_i = \Phi \boldsymbol{y}_i$.
2. Cluster the sampled space $\{\boldsymbol{\xi}_1 \dots \boldsymbol{\xi}_M\}$.
3. For each vector $\boldsymbol{\xi}_i$ in cluster $\mathcal{C}_m$, reconstruct the original vector $\boldsymbol{x}_i$ using equation 1.
4. Compute each cluster definition in the unsampled space based on the cluster vectors $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_M\}$.
5. Re-assign each unsampled vector based on the unsampled cluster definitions.

The cluster definition is what defines the cluster (e.g. a hyperplane, a point in the space, a direction in the space). We also make use of the cluster definition when assigning new vectors to a cluster. Note that step 2 can be performed using compressive sampling reconstruction, or by accessing the original data if possible.

Using this algorithm, we are able to perform the clustering in the reduced space, requiring less computation and less memory. The algorithm requires:

- one pass over the data to sample each vector which is passed to the clustering method.
- a second pass over the data to compute each cluster definition.
- a third pass over the data to assign each vector to a cluster based on the cluster definitions.

Therefore, rather than requiring many scans of high dimensional data set, we are able to cluster the data in a reduced space requiring only three scans of the high dimensional data.

3.1 Document Clustering

The common representation of documents as vectors uses a vector space where each dimension represents a unique term in the document collection. For example, a document existing in a document collection containing 200,000 unique terms, is represented using a 200,000 dimensional vector. We can see that when using this vector space, the dimension of the vector space is related to the number of documents in the collection, and can only grow as new documents are added.

Using this vector representation also leads to very sparse document vectors (containing many zero elements). If a document contains 100 unique terms, then its associated document vector in the 200,000 dimensional space would contain 199,900 zero elements. It is this feature that leads to high compression ratios when constructing a document index for information retrieval.

It is common for less that 1% of document vector elements to contain nonzero values. Therefore it is safe to assume that document vectors are a sparse representation of the documents.

Document vector elements contain the frequency (or weighted frequency) of the associated term in the associated document. Based on this, it does not make

sense to use Euclidean k-means to cluster the document set, as Euclidean distance is a geometric distance between two points. The value of zero in a vector is simply a position of a point to the Euclidean distance metric. In our case, zero is not a position, but an absence of a term. Therefore, we provide a modification of the k-means algorithm using the angle between two vectors as a measure of similarity.

3.2 Complex Radial K-means

In this section, we present a form of k-means that uses the angle between vectors as a measure of similarity, and is able to cluster vectors existing in a complex vector space.

K-means is an iterative process that records a cluster centre vector $\boldsymbol{c}_m$ for each cluster and adjusts it until stability is reached.

The first part of any k-means process is the initialisation of the centre vectors. In this article, we take the approach of Hartigan and Wong [8], and adapt it for use with the vector angle similarity metric. Given the mean vector $\bar{\boldsymbol{x}} = \sum_i \boldsymbol{x}_i$ and the angle between each vector and the mean vector $\theta_i = \bar{\boldsymbol{x}} \angle \boldsymbol{x}_i$, the cluster centres are initialised using:

$$\boldsymbol{c}_m = \boldsymbol{x}_{\text{round}(1+(m-1)N/C)}$$

where the vectors $\boldsymbol{x}_i$ are sorted by θ_i (the angle θ_1 associated to $\boldsymbol{x}_1$ is the smallest and the angle θ_N associated to the vector $\boldsymbol{x}_N$ is the largest angle), and round(x) rounds fractional values to their nearest integer.

Using this initialisation, we can compute the radial k-means clusters using the iterative process:

$$\begin{aligned}
\boldsymbol{c}_m &\leftarrow \sum_{i \in \mathcal{C}_m} \boldsymbol{x}_i \text{ for } m \in \{1, \ldots, C\} \\
\mathcal{C}_m &\leftarrow \{x_i | m = \underset{m}{\text{argmin}} \left(\boldsymbol{c}_m \angle \boldsymbol{x}_i\right)\} \text{ for } m \in \{1, \ldots, C\}
\end{aligned}$$

where $\mathcal{C}_m$ is the set of vectors associated to cluster m.

We define the angle between any two complex vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as:

$$\cos\left(\boldsymbol{x}_i \angle \boldsymbol{x}_j\right) = \frac{\text{Re}(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle)}{\|\boldsymbol{x}_i\|_2 \|\boldsymbol{x}_j\|_2}$$

where $\|\boldsymbol{x}_i\|_2 = \left(\sum_j |x_{i,j}|^2\right)^{1/2}$ and $\text{Re}(a + ib) = a$ for real values a and b.

3.3 Approximate k-means Document Clustering

By using radial k-means as our clustering algorithm, in conjunction with our compressive clustering algorithm, we obtain a compressive clustering algorithm suitable for documents:

1. Sample the features of the vector space using the sampling function Φ.
2. Perform k-means on the sampled space.
3. For each vector $\boldsymbol{\xi}_i$ in cluster $\mathcal{C}_m$, reconstruct the original vector $\boldsymbol{x}_i$.
4. Compute the cluster centre $\boldsymbol{c}_m = \sum_{i \in \mathcal{C}_m} \boldsymbol{x}_i$.
5. Compute the unsampled space clusters using the unsampled space cluster centres.

To complete the compressive clustering algorithm we must choose an appropriate transform and sampling matrices. The transform matrix Ψ provides the relationship between the sparse domain and the data domain. We have stated in section 3.1 that the document vectors are sparse, therefore a suitable transform matrix is the identity matrix ($\Psi = I$).

The sampling matrix must be incoherent to the transform matrix, therefore the perfect choice is a random sample of the discrete Fourier basis:

$$\xi_{j,k} = \sum_{n=1}^{N} y_{j,n} e^{-i2\pi(k-1)(n-1)/N}$$

where $y_{j,n}$ is the nth element of vector $\boldsymbol{y}_j$, and $\xi_{j,k}$ is the kth Discrete Fourier Transform (DFT) sample of vector $\boldsymbol{y}_j$. Noting that the discrete Fourier basis is complex, we will also examine the discrete Cosine basis:

$$\xi_{j,k} = \sum_{n=1}^{N} y_{j,n} \cos\left(\pi(k-1)(2n-1)/2N\right)$$

where in this case $\xi_{j,k}$ is the kth Discrete Cosine Transform (DCT) sample of the vector $\boldsymbol{y}_j$.

Given a transformation matrix Φ and vectors $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$, we can compute the transformed vectors: $\boldsymbol{\xi}_i = \Phi \boldsymbol{y}_i$ and $\boldsymbol{\xi}_j = \Phi \boldsymbol{y}_j$. If we define $\boldsymbol{\xi}_i \angle \boldsymbol{\xi}_j$ as the angle between vectors $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j$, then:

$$\begin{aligned}
\cos\left(\boldsymbol{\xi}_i \angle \boldsymbol{\xi}_j\right) &= \frac{\langle \boldsymbol{\xi}_i, \boldsymbol{\xi}_j \rangle}{\|\boldsymbol{\xi}_i\|_2 \|\boldsymbol{\xi}_j\|_2} \\
&= \frac{\langle \Phi \boldsymbol{y}_i, \Phi \boldsymbol{y}_j \rangle}{\|\Phi \boldsymbol{y}_i\|_2 \|\Phi \boldsymbol{y}_j\|_2} \\
&= \frac{\boldsymbol{y}_i^T \Phi^T \Phi \boldsymbol{y}_j}{\sqrt{\boldsymbol{y}_i^T \Phi^T \Phi \boldsymbol{y}_i} \sqrt{\boldsymbol{y}_j^T \Phi^T \Phi \boldsymbol{y}_j}}
\end{aligned}$$

From this expansion, we can clearly see that if Φ is a unitary transformation, then $\cos\left(\boldsymbol{\xi}_i \angle \boldsymbol{\xi}_j\right) = \cos\left(\boldsymbol{y}_i \angle \boldsymbol{y}_j\right)$.

The Discrete Fourier transform and Discrete cosine transform are unitary transformations, implying that vector norms are preserved when the transformation is performed. By sampling the rows of the transformation, we obtain a matrix that is no longer unitary, but has been shown to be approximately orthogonal [5]:

$$(1-\delta)\|\boldsymbol{y}\|_2^2 \leq \|\Phi \boldsymbol{y}\|_2^2 \leq (1+\delta)\|\boldsymbol{y}\|_2^2$$

for small values of δ.

Therefore, if Φ is approximately orthogonal, we can show that $\cos(\boldsymbol{\xi}_i \angle \boldsymbol{\xi}_j) \approx \cos(\boldsymbol{y}_i \angle \boldsymbol{y}_j)$. It is due to this property that our clustering in the reduced space provides a good approximation to the clustering that would be obtained in the unsampled high dimensional space.

This compressive clustering algorithm using complex k-means clustering can be considered a type of sketching algorithm as investigated in [9], where, in our case, we are producing compact vectors that attempt to preserve the original angle between each vector.

4 Performance

We have seen that the iterative process of k-means constantly requires access to the vector data. Therefore, an efficient process would store the vector set in memory to avoid disk accesses. Given this constraint, the size of the data set that we can process depends on the size of the accessible memory.

In this section, we will examine the performance of radial k-means on a small document set from the SMART collection[1]. The document set contains the document set CRAN, CACM, CISI and MED containing 1398, 3204, 1460, and 1033 documents respectively, totalling $7,095$ documents with $14,523$ unique terms.

To reduce the size of the vector space, we ignored all terms that appeared in only one document. By ignoring these terms, we will not affect the results since they do not affect to the similarity between any two documents. This reduced the number of terms from $14,523$ to $7,866$.

4.1 Cluster Accuracy

We computed the accuracy of each cluster using the Jaccard coefficient:

$$J(\mathcal{C}_m, \mathcal{C}_n) = \frac{|\mathcal{C}_m \cap \mathcal{C}_n|}{|\mathcal{C}_m \cup \mathcal{C}_n|}$$

where $\mathcal{C}_m$ and $\mathcal{C}_n$ are the sets of vectors associated to clusters m and n respectively, and $|\cdot|$ is the cardinality operator.

For each experiment performed, we obtained a data set with a recommended clustering. To compute the accuracy, we compared the computed clustering to the recommended clustering and chose the permutation of clusters that optimised the following function:

$$\rho = \underset{p}{\operatorname{argmax}} \left(\sum_{i=1}^{C} J(\mathcal{C}_{p_i}, \mathfrak{C}_i) \right)$$

where $\mathfrak{C}_i$ is the recommended cluster set i, $\mathcal{C}_{p_i}$ is the computed cluster set p_i, p_i is the ith value in permutation p, and ρ is the best matching cluster permutation when compared to the recommended cluster set.

[1] ftp://ftp.cs.cornell.edu/pub/smart

4.2 Radial K-means without Sampling

As a baseline measure, we will apply the k-means algorithm to the document vector space without sampling. Doing so produced the results shown in table 1.

Table 1. Cluster accuracy of radial k-means on the whole feature space (no sampling performed). The small size of the document collection allows us to load its contents into memory and perform k-means. We can see that it produces high accuracy, and takes nearly seven minutes to complete.

Accuracy				Time (sec)
CRAN	CACM	CISI	MED	
0.9709	0.9408	0.8798	0.9778	409.82

We can see from the results that radial k-means has successfully partitioned the document collection into the individual document sets, with only the CISI document set accuracy being below 0.9. We can also see that the algorithm took 409.82 seconds to complete (nearly 7 minutes).

4.3 Radial K-means with DFT Sampling

We will now apply the compressive clustering algorithm from section 3.3 to our document collection and compare its accuracy to the clustering found without sampling.

Our first set of results used a sample of the DFT basis as the sampling matrix Φ. We computed results using sample sizes of 16, 32, 64, 128, 256, 512 and 1024. Since the samples are randomly selected we ran 10 trials using each sample size and have reported the mean accuracy in Table 2 and the standard deviation of the accuracy in Table 3.

We can see from the mean accuracy results that there is a clear increase in accuracy as the number of random DFT features used increases, and that the increase saturates at 256 features. We can see that the accuracy for samples of 256 or more are very close to the unsampled clusters obtained in Table 1, being only 0.1 or 0.2 in difference.

Table 2 also contains timing information, showing the mean time taken to run the compressive clustering algorithm. We can see that the times are much shorter than the 409.82 seconds taken without sampling, with 256 DFT features being 14 times faster.

The table of standard deviations (Table 3) provide us with an understanding of the effect of the random sampling. It is interesting to see that the standard deviation is much greater for sampled DFT features of 128 and less when compared to those of 256 and greater. This implies that a low standard deviation in a set of clustering results gives an indication that we have taken enough samples to provide a close approximation to the unsampled clusters. Therefore the cluster standard deviation could easily be used as a measure of the compressive cluster accuracy.

Table 2. Cluster accuracy when using compressive clustering with the Discrete Fourier Transform (DFT). The number of randomly sampled DFT features are provided in the first column, with the accuracy of each cluster (measured using the Jaccard coefficient) in the following four columns, followed by the computation time in seconds.

DFT Feat	Mean Accuracy				Time (sec)
	CRAN	CACM	CICI	MED	
1024	0.9599	0.9196	0.8528	0.9620	74.11
512	0.9258	0.9403	0.8531	0.8944	36.30
256	0.9688	0.9328	0.8607	0.9638	28.49
128	0.8882	0.7117	0.6093	0.5981	24.66
64	0.8219	0.8498	0.7652	0.7184	25.70
32	0.6430	0.7304	0.5976	0.4092	34.93
16	0.7091	0.5815	0.4403	0.3331	24.95

Table 3. Cluster accuracy standard deviation when using compressive clustering with the Discrete Fourier Transform (DFT). The number of randomly sampled DFT features are provided in the first column, with the standard deviation of the accuracy of each cluster (measured using the Jaccard coefficient) in the following four columns.

DFT Feat	Accuracy Standard Deviation			
	CRAN	CACM	CICI	MED
1024	0.0367	0.0670	0.0822	0.0408
512	0.1229	0.0037	0.0660	0.2071
256	0.0028	0.0148	0.0248	0.0132
128	0.1345	0.2112	0.2425	0.3925
64	0.1573	0.1079	0.0912	0.2704
32	0.1844	0.1683	0.1337	0.2357
16	0.1492	0.1412	0.1018	0.1982

4.4 Radial K-means with DCT Sampling

The use of the DFT feature samples requires us to work with complex numbers, meaning that the chosen clustering algorithm also has to cluster complex numbers. Since many clustering clustering algorithms are designed for real numbers only, we will examine a method of sampling real values. Therefore we will examine the effect of DCT feature samples on the accuracy of our compressive clustering algorithm.

Our next set of results used a sample of the DCT basis as the sampling matrix Φ. We computed results using sample sizes of 16, 32, 64, 128, 256, 512 and 1024. Since the samples are randomly selected, we ran 10 trials using each sample size and have reported the mean accuracy in Table 4 and the standard deviation of the accuracy in Table 5.

Table 4. Cluster accuracy when using compressive clustering with the Discrete Cosine Transform (DCT). The number of randomly sampled DCT features are provided in the first column, with the accuracy of each cluster (measured using the Jaccard coefficient) in the following four columns, followed by the computation time in seconds. The superscript dagger (†) denotes a significant difference when compared to the same result using the DFT. The subscript star (∗) denotes a significant difference when compared to the associated result when using the DFT with half the number of features.

DCT Feat	Mean Accuracy				Time (sec)
	CRAN	CACM	CICI	MED	
1024	0.9494	0.8895	0.8022	0.9280	65.10_{*}
512	$0.9139^{\dagger}_{*}$	0.9207	0.8450	0.8977	48.93_{*}
256	$0.8752^{\dagger}$	0.8989_{*}	$0.7804^{\dagger}$	$0.8006^{\dagger}$	31.54_{*}
128	$0.7423^{\dagger}$	0.7662	0.5652	0.4937	$35.33^{\dagger}_{*}$
64	0.7163	0.7442	$0.5867^{\dagger}$	0.5109	29.78
32	0.5669	$0.5437^{\dagger}$	$0.4197^{\dagger}$	0.3207	27.14
16	$0.4694^{\dagger}$	0.5413	0.4074	0.3019	25.95

From first glance, the accuracy values in Table 4 are lower than those of the DFT in Table 2, but we must remember that shown accuracies are mean values of the ten sample runs for each feature set size. The random sampling of features implies that the results from each experiment will lead to different clustering accuracies. Therefore we employ the use of statistical significance testing to assist us in understanding if there actually is a difference in the clustering, or if the difference is purely by chance.

In each of our experiments, we compute the clustering accuracy obtained using ten different sets of randomly sampled features. Therefore, we have ten different clustering accuracy measurements for each experiment. The statistic that we want to test is if the true mean accuracy of the clustering using the DCT features is different to the true mean accuracy of the clustering using the DFT features.

Since we have only ten samples (ten clustering results) for each method, we cannot have confidence in the t-test, therefore we will employ the use of Wilcoxon rank sum test.

Table 4 contains the results from the significance tests in the form of a superscript dagger (†). A dagger on the accuracy denotes that there is a significant difference at the 0.05 level. We can see that there are four cases that the CRAN cluster is significantly different, three cases where the CISI cluster is significantly different and one case where CACM and MED clusters were significantly different. From examination, we can see that the DCT sampling is worse in all of these cases.

If we consider that the DFT features are complex (containing real and imaginary portions) and that the DCT features are only real, it would be a fairer comparison if we compared the DCT feature results to those of the DFT with half the number of features. For example, if 256 DCT features were selected from

a vector, they would occupy the same memory as 128 DFT features, since each DFT feature contains two values (a real and imaginary value).

Table 4 contains the results from the significance tests comparing each DCT feature sample accuracy to each DFT feature sample of half size, in the form of a subscript star ($*$). A star on the accuracy denotes that there is a significant difference at the 0.05 level. We can see that there are now only two cases in total where there is a significant difference, implying that DCT features produce similar results to DFT features when half of the DFT feature samples are used.

Table 4 also contains timing information, showing the mean time taken to run the compressive clustering algorithm. We can see that similar to the DFT sampling, the times are much shorter than the 409.82 seconds taken without sampling, with 512 DCT features being 8 times faster.

Significance tests were also performed on the times to find if the time taken to perform compressive clustering using DCT features was significantly different to the times taken to perform compressive clustering using DFT features. The times with a superscript dagger show a significant difference at the 0.05 level when comparing experiments with the same number of features, while times with subscript star show a significant difference in times at the 0.05 level when comparing experiments using DCT features with those using half the number of DFT features. We can see that there is only one case where a significant difference in time is shown when comparing DCT and DFT with the same number of feature samples. This implies that the use of complex numbers has not increased the time of the clustering. If we examine the subscript stars, we can see that there are four times when there is a significant difference between the time for DCT based compressive clustering and DFT based compressive clustering, where the DCT based method produces longer times.

Table 5. Cluster accuracy standard deviation when using compressive clustering with the Discrete Cosine Transform (DCT). The number of randomly sampled DCT features are provided in the first column, with the standard deviation of the accuracy of each cluster (measured using the Jaccard coefficient) in the following four columns.

DCT Feat	Accuracy Standard Deviation			
	CRAN	CACM	CICI	MED
1024	0.0454	0.1102	0.1640	0.0961
512	0.0699	0.0500	0.0669	0.0972
256	0.1586	0.0952	0.1198	0.2903
128	0.1986	0.1730	0.2540	0.3653
64	0.1975	0.1190	0.2045	0.3012
32	0.1508	0.1626	0.1518	0.2227
16	0.0930	0.1615	0.2737	0.1638

The table of standard deviations (Table 5) provide us with an understanding of the effect of the random sampling. We can see that the table is similar to Table 3 except that the drop in standard deviation does not occur until 512 DCT features are sampled. This re-enforces our belief that the standard deviation can be used to measure the accuracy of our approximation to the unsampled clustering.

5 Clustering Large Scale Document Sets

Our final set of experiments examine the use of compressive clustering on a much larger document set. Just as in the previous section, we have again taken a set of document collections and combined them. The document collections are a set of newspaper articles from the Associated Press (AP), articles from the Financial Review (FR), articles from the Wall Street Journal (WSJ) and articles from Ziff Publishing (ZIFF). These document collections are available on disk two of the TIPSTER collection[2] and were used at TREC-1 to 5. Individually, they contain 79,919, 19,860, 74,520, and 56,920 documents respectively, totalling to 231,219 documents. This document collection contains 208,932 terms. Again, we removed all terms that appeared in only one document, which reduced the term count to 108,734 terms.

Table 6. Cluster accuracy for radial k-means on the whole feature space (no sampling performed) for the large document set (231,219 documents). These results are used as a baseline for the compressive clustering results in Table 7.

Accuracy				Time (sec)
AP	FR	WSJ	ZIFF	
0.6780	0.9962	0.5969	0.8603	181734

We have computed the radial k-means clusters for the large document set (without sampling) as a baseline for the compressive clustering results. The clustering results are shown in Table 6. We can see that the radial k-means process clustered most of the FR and ZIFF documents into correct clusters, but there was confusion amongst the AP and WSJ documents. We can also see that the process took over 50 hours to complete.

To continue the experiment, we ran our compressive clustering algorithm with both the DFT sampling and DCT sampling, and the compressive clustering algorithm. The number of sampled features was chosen as the largest that we could store in memory. For the DFT sampling this was 128 features, while for the DCT features this was 256 features (due to the DFT features being complex and DCT features being real). The recommended clustering for this document

[2] http://trec.nist.gov/data/docs_eng.html

Table 7. A comparison of compressive sampling using DFT sampling and DCT sampling. The first two columns provide the sampling transform and the number of features sampled, the following four columns provide the cluster accuracy in terms of the Jaccard coefficient, and the last three columns provide the computation time of the stages involved. Note that the DFT sampling uses half the number of samples since each sample is complex, containing a real and imaginary component.

Transform	Features	Accuracy				Computation Time (sec)			
		AP	FR	WSJ	ZIFF	Pass 1	k-means	Pass 2	Pass 3
DFT	128	0.6704	0.9959	0.5821	0.8617	3626	458	2398	2334
DCT	256	0.6839	0.9964	0.6134	0.8627	3785	1130	2412	2342

collection is each cluster containing only documents from one document set (e.g. cluster 1 contains only documents from the AP document set). The accuracy and timing results are provided in Table 7.

We can see in Table 7 that we obtained high accuracy clusters of the Financial Review and Ziff Publishing articles, implying that the clusters found by k-means was very similar to those in the corresponding collections. The reduction in accuracy for the Associated Press and Wall Street Journal collections shows that there was confusion for a set of the articles as to which cluster they belonged to. The Associated Press and Wall Street Journal collections both contain newspaper articles, therefore there also may be similarity in the content that they each publish. From the results it is obvious that the k-means algorithm has computed similarity between articles from each of the AP and WSJ sources and hence caused a reduction in accuracy.

When comparing the accuracy of DFT and DCT features, Table 7 shows that there is little difference, with the DCT sampled compressive clustering providing slightly greater accuracy, but there is no evidence that one method is better than the other.

The computation time section of Table 7 provides us with a break down of the computation time of each stage in the algorithm. Pass 1 involves scanning through the data set and performing either DFT or DCT feature sampling. The only difference between the two is the time of performing the transform sampling. We can see that the times are similar for both DFT and DCT sampling, implying that the transform sampling times are also the same. The k-means time shows the number of seconds spent computing the radial k-means clusters. It clear that the DFT feature sampled data was much faster to process for k-means than the DCT based data. This is interesting, since as we have said, the DFT features are complex, making the number of integers processed the same as processed in the DCT based features. But the time taken to perform k-means is on the DFT features is less than half of the time to perform k-means on the DCT feature samples.

Pass 2 and 3 involve computing the cluster centres in the unsampled space and recomputing the document assignments to each cluster. We can see that

these are both similar and that they are independent of the feature sampling method used.

If we compare the compressive clustering results in Table 7 to the clustering results in Table 6, we find that the accuracy results are very similar. This implies that the compressive clustering algorithm is providing an excellent estimate of the clusters. If we compare the times, we find that the compressive clustering algorithm took approximately 2.5 hours to compute the clusters (for the DFT and DCT methods), while clustering the original data took over 50 hours. This demonstrates the benefit of compressive clustering.

From this we can see that compressive clustering is an exciting new method of computing clusters in high dimensional data. In this article we have explored its application to document clustering, but we can see that with correctly chosen transform and sampling matrices, we can apply compressive clustering to any large scale clustering problem to produce clusters that would have otherwise been not computable.

6 Related Work

The work we have presented is related to Locality-Sensitive Hashing (LSH) [6], and Random projection [2], where in our case the hashing/projection is defined by sampling from the most incoherent linear transformation to the sparse feature space, which in the case of text document vectors, is the Fourier transform or Cosine transform.

Note that when using compressive sampling for dimension reduction and clustering, we are able to reconstruct any vector in the reduced space to its sparse equivalent in the original feature space under certain conditions. This is not possible when performing LSH or Random projection.

7 Conclusion

Clustering large scale data sets requires intense computation and large storage space (such as memory and disk space). The clustering process itself requires continuous access to the data and therefore we benefit from storing the data in memory to avoid lengthy disk accesses.

Compressive sampling is a new concept of Information theory that allows us to sample a small number features from a data set such that we are able to perfectly reconstruct the data, with knowledge of the sampling scheme.

In this article, we presented compressive clustering, an algorithm which utilises the sampling of compressive sampling to obtain a reduced feature space. Using this reduced space, we are able to obtain a close approximation to the clustering that would be obtained on the unsampled data. Using the algorithm presented in this article, compressive clustering can be applied to any domain to obtain approximate clusters in high dimensional data, given appropriate sampling and transform matrices.

We applied compressive clustering using radial k-means to two document collections. We showed that compressive clustering using discrete Fourier and discrete Cosine sampling provided a close approximation to the clusters computed on the unsampled data in 1/14th of the time on the first document set. On the second larger document set containing 231,219 documents, we showed that compressive clustering can provide a very accurate clustering in 1/20th of the time. This shows that compressive clustering is a very useful tool that can be used to quickly compute approximate clusters.

References

1. Baraniuk, R., Davenport, M., DeVore, R., Wakin, M.: A simple proof of the restricted isometry property for random matrices. Constructive Approximation 28, 253–263 (2008), doi:10.1007/s00365-007-9003-x
2. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 245–250. ACM, New York (2001)
3. Candes, E.J., Tao, T.: Decoding by linear programming. IEEE Transactions on Information Theory 51(12), 4203–4215 (2005)
4. Candes, E.J., Wakin, M.B.: An introduction to compressive sampling. IEEE Signal Processing Magazine 25(2), 21–30 (2008)
5. Candes, E.J.: Compressive sampling. In: Proceedings of the International Congress of Mathematicians. European Mathematical Society, Madrid (2006)
6. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: Proceedings of the 25th International Conference on Very Large Data Bases, VLDB 1999, pp. 518–529. Morgan Kaufmann Publishers Inc., San Francisco (1999)
7. Goyal, V.K., Fletcher, A.K., Rangan, S.: Compressive sampling and lossy compression. IEEE Signal Processing Magazine 25(2), 48–56 (2008)
8. Hartigan, J.A., Wong, M.A.: Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics) 28(1), 100–108 (1979)
9. Li, P., Church, K.W., Hastie, T.J.: Conditional random sampling: A sketch-based sampling technique for sparse data. In: Advances in Neural Information Processing Systems, vol. 19, p. 873 (2007)

Ancestor Relations in the Presence of Unobserved Variables

Pekka Parviainen and Mikko Koivisto

Helsinki Institute for Information Technology HIIT, Department of Computer Science
University of Helsinki, Finland
{pekka.parviainen,mikko.koivisto}@cs.helsinki.fi

Abstract. Bayesian networks (BNs) are an appealing model for causal and non-causal dependencies among a set of variables. Learning BNs from observational data is challenging due to the nonidentifiability of the network structure and model misspecification in the presence of unobserved (latent) variables. Here, we investigate the prospects of Bayesian learning of ancestor relations, including arcs, in the presence and absence of unobserved variables. An exact dynamic programming algorithm to compute the respective posterior probabilities is developed, under the complete data assumption. Our experimental results show that ancestor relations between observed variables, arcs in particular, can be learned with good power even when a majority of the involved variables are unobserved. For comparison, deduction of ancestor relations from single maximum a posteriori network structures or their Markov equivalence class appears somewhat inferior to Bayesian averaging. We also discuss some shortcomings of applying existing conditional independence test based methods for learning ancestor relations.

1 Introduction

Directed acyclic graphs (DAGs) provide a convenient formalism for representing relationships among a set of variables in terms of *conditional independencies* (CIs) [17,18]. To enable quantitative reasoning, a DAG is often attached to a probability measure that obeys exactly the CIs represented by the DAG. While the probability measure alone would of course suffice for probabilistic inference on the variables, the DAG contains additional structure that supports particularly causal interpretations: an arc between two variables represents a direct cause–effect relationship. The pair of the DAG and the measure is sometimes called a *Bayesian network*; the modifier "Bayesian" suggests a degree-of-belief interpretation of probability, which is applicable also when, for instance, the causal mechanisms are believed to be deterministic but just unknown to the modeller. Often a single Bayesian network is used for simultaneous modelling of several "similarly behaving" vectors of variables; then a node of the DAG corresponds to several random variables that are often treated as observations. If the nodes are observed, that is, the values of the respective random variables are known, standard principles of statistical inference can be implemented to derive more or less uncertain conclusions about the Bayesian network model, especially the DAG.

While automatic construction, or *learning*, of such DAGs from observational data is desirable, the task is notoriously challenging. First, a set of CIs can be represented

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 581–596, 2011.

by a number of different DAGs that form a so-called Markov equivalence class. Thus, the assumed "data generating DAG" cannot be identified by the represented CIs only. Second, if there are unobserved nodes at work, it may be that no DAG on the observed nodes can represent exactly the CIs among them. Then, the DAG model is misspecified in a way that directly affects the end result of statistical inference: the DAG. Third, the combinatorial and constrained nature of the DAG model brings major challenges concerning modeling complexity and, in particular, computational complexity.

To address these challenges, the art of learning DAGs from data has been developed in two rather distinct directions. Constraint-based methods [17,18] rely on testing CIs. While the approach is not particularly suitable for importing prior knowledge, nor for efficient use of data, nor for managing nonidentifiability issues, it has given arise to a profound theory for dealing with unobserved variables. On the other hand, score-based methods [1,11], particularly Bayesian ones [9,15], excel in flexibility and statistical efficiency in the translation of what was known prior the observations to what is known *a posteriori*, including a proper treatment of nonidentifiability. For example, in the Bayesian approach there is no need to infer a single *maximum a posteriori* (MAP) DAG or its Markov equivalence class when there are many other almost equally good DAGs—instead, one may report *structural features*, e.g., arcs, that have a high posterior probability. As a drawback, it seems difficult to extend the Bayesian approach to handle the issue of unobserved nodes in a computationally efficient manner. Indeed, the score-based methods are often applied ignoring unobserved nodes altogether: either one refuses to make any conclusions, especially causal, about the DAG; or, one makes such conclusions at an unquantified risk of erroneous claims. While there are some notable exceptions that employ various score-driven heuristics to discover unobserved nodes [3,4,5,8], principled methods are yet to be developed.

Motivated by these concerns, this paper investigates the potential of Bayesian learning of structural features of DAGs *on the observed nodes only*. Are there structural features that can be reliably learned from observational data, even if there may be some unobserved nodes at work? We find this question highly relevant and interesting, since the popular score-based methods for structure learning ignore unobserved nodes, which, however, are expected to be present in typical practical scenarios.

As a natural candidate for such a structural feature we consider *ancestor relations*. A node s is an ancestor of another node t if there is at least one directed path from s to t. An arc from s to t can be viewed as a special case of ancestor relations. The idea of learning ancestor relations from data is, of course, not new. Spirtes et al. [19] investigate constraint-based learning of ancestor relations using the FCI algorithm in a small-case empirical study. Their results suggest that reliable learning of ancestor relations is possible in the presence of unobserved nodes; however, direct comparison to our methods is not reasonable, as the predictions by FCI are unquantified and predictions are not necessarily made for all pairs of nodes. A Bayesian treatment is given by Friedman and Koller [9]: under the supposition that there are no unobserved nodes, DAGs are sampled (via node orderings) from their posterior distribution using a Markov chain Monte Carlo simulation and the posterior probabilities of ancestor relations, also called path features, are estimated based on the sampled DAGs; based on the posterior

probabilities, the ancestor relation is either claimed to hold or not to hold, potentially depending on the relative costs of making incorrect positive or negative claims.

Our present work contributes to this line of research by (a) giving a dynamic programming algorithm that computes the *exact* posterior probabilities of ancestor relations and by (b) studying the statistical power of learning such relations in the presence of *unobserved nodes*. From a computational point of view, ancestor relations present a new algorithmic challenge, as they do not fall in the class of modular features [9,15]; see also a recent discussion by Tian et al. [21]. As can be expected, the computational complexity of the exact algorithm is exponential; the algorithm runs in $O(3^n n^2)$ time and $O(3^n)$ space on n-node instances. While such exponential complexity, of course, renders the algorithm computationally feasible only for relatively small instances, one should note that such moderately exponential time algorithms, that is, algorithms whose base constant is quite small, have attracted substantial interest in the context of Bayesian networks; see, e.g., Tian and He [20] and Kang et al. [12]. Both our algorithm and the power study assume that the prior over DAGs is of a restricted form, namely order-modular in the sense of Koivisto and Sood [15]; see also Friedman and Koller [9]. An order-modular prior generally assigns different prior probabilities to different DAGs within a Markov equivalence class. Compared to the uniform prior, this is, however, neither a disadvantage nor an advantage in general (besides the computational advantage), since the modeller's subjective prior may well be better represented with an order-modular prior than with the uniform prior. We also stress that, while our approach is fully Bayesian, the model is misspecified (does not fully represent the modeller's beliefs regarding unobserved nodes). Thus, the present work should be viewed as a study of the robustness of Bayesian averaging to model misspecification.

The remainder of the paper is structured as follows. We begin in Section 2 by reviewing a modular Bayesian network model [9,15]. Then we give a dynamic programming algorithm for exact computation of the target posterior probabilities. Section 3 reports on empirical results concerning the statistical efficiency of learning ancestor relations and directed or undirected arcs with a varying number of observed nodes and data points per node. As an obvious (heuristic) alternative to Bayesian averaging, we also consider the deduction of ancestor relations from single MAP DAGs or their Markov equivalence classes. We also report on and discuss results obtained by the constraint-based algorithm, FCI [19] Finally, we summarize in Section 4.

2 Bayesian Discovery of Ancestor Relations

Our Bayesian network model relates a DAG on n nodes with m random variables per node (often treated as the observations or data; see below) by defining a joint probability measure on them.[1] Without any loss in generality we let the node set be $N = \{1, 2, \ldots, n\}$ and identify a DAG with its arc set $A \subseteq N \times N$; the set of *parents* of node v is $A_v = \{u : uv \in A\}$. If a DAG contains a directed path from u to v, then u is called an *ancestor* of v, and v a *descendant* of u. With each node v we associate a

[1] Note that while the m variables will be independent and identically distributed given a fully specified model, a Bayesian model also includes priors over the parameters and operates on exchangeability, not on independence.

sequence of random variables $D_v = D_{v1}D_{v2}\cdots D_{vm}$; we write D for $D_1D_2\cdots D_n$. A joint probability measure $p(A, D)$ is composed as $p(A, D) = p(A)p(D|A)$ with the following structure. By standard interpretation of conditional independencies on a DAG we have

$$p(D|A) = \prod_{v \in N} p(D_v|D_{A_v}, A_v) \; ;$$

for our purposes it is irrelevant how the local conditional measures $p(D_v|D_{A_v}, A_v)$ are further specified. For computational convenience, we define an *order-modular* prior for the DAG A. To this end, the joint prior of the DAG A and a linear order $L \subseteq N \times N$ on N is specified by

$$p(A, L) = \prod_{v \in N} \rho_v(L_v)q_v(A_v) \; ,$$

where $L_v = \{u : uv \in L\}$ consists of the predecessors of v in L and ρ_v and q_v are non-negative functions. The prior for the DAG is obtained by marginalizing the joint prior, that is, $p(A) = \sum_{L \supseteq A} p(A, L)$. Note that the sum is over all topological orderings of the DAG and that the set inclusion notation is valid (L is a superset of A). Note also that in practice the functions need to be specified only up to some normalization constant, e.g., $\rho_v(L_v) \propto 1$ and $q_v(A_v) \propto 1/\binom{n-1}{|A_v|}$, as the normalization constant will cancel in the quantities of our interest.

We consider a setting where the values of D, called the *data*, are observed, and we are interested in the posterior probability that the DAG A contains some specified structural feature. We will focus on two kinds of events that relate two nodes: uv is an arc in A, denoted $u \rightarrow v$; s is an ancestor of t in A, denoted $s \rightsquigarrow t$.

2.1 Computation

From an algorithmic point of view, it is convenient to compute the posterior probability of a structural feature $f(A)$ given the data D as the ratio $p(f(A), D)/p(D)$. Letting f be a 0–1-valued indicator function, we have $p(f(A), D) = \sum_A f(A)p(D|A)p(A)$, where the sum is over all DAGs on N. Koivisto and Sood [15] show that if $f(A)$ factorizes into a product of family-wise indicators $f_v(A_v)$, then the probabilities can be computed by dynamic programming (DP) across the node subsets of N in time $O(n^2 2^n)$ and space $O(n2^n)$; furthermore, the arc events $u \rightarrow v$ can be handled simultaneously for all the $n(n-1)$ node pairs uv within the same bounds [14].

The computation of the posterior probabilities of ancestor–descendant relationships seems more challenging, as the existence of directed path between two fixed nodes is a global property that does not factorize into independent local properties. We next give a DP algorithm that for every node subset S computes its contribution to the target probability, $p(s \rightsquigarrow t, D)$, assuming the nodes in S are the first $|S|$ nodes in the linear order L; the contribution is a sum over all possible DAGs, A_S, on the node set S. The key difference to the existing DP algorithms for arc probabilities or for the maximum posterior probability is that, aside from the set S, we need to keep a handle on the nodes in S that are descendants of the source node s. To this end, we define a set $T \subseteq S$ such

that $t \in T$ if and only if s is an ancestor of t or $t = s$. Thus, every DAG on S determines exactly one such set $T \subseteq S$.

Furthermore, for sets S and $T \subseteq S$ and a linear order $L_S \subseteq S \times S$ on the respective node set $S \subseteq N$, we use the shorthand

$$\mathcal{A}(L_S, S, T) = \{A_S \subseteq L_S : \forall v \in S \, (s \rightsquigarrow v \text{ in } A_S \text{ iff } v \in T)\} ;$$

in words, $\mathcal{A}(L_S, S, T)$ contains a particular DAG A_S on S if and only if A_S is compatible with L_S and A_S contains a path from s to every node $v \in T$, and not to any other node in S.

Our dynamic programming algorithm will compute a function $g_s(S, T)$, defined for all $S \subseteq N$ and $T \subseteq S$ by

$$g_s(S, T) = \sum_{L_S} \sum_{A_S \in \mathcal{A}(L_S, S, T)} \prod_{v \in S} \rho_v(L_v)\beta_v(A_v) ,$$
$$\beta_v(A_v) = q_v(A_v)p(D_v | D_{A_v}, A_v) ,$$

where the outer summation is over all linear orders L_S on S. Intuitively, $g_s(S, T)$ is the sum of $p(A, D, L)$ over all DAGs A_S and linear orders L, with $A_S \subseteq L$, such that S are the first nodes in the order L and there is a path from s to $v \in S$ in A_S if and only if $v \in T$. That the values $g_s(S, T)$ are sufficient for computing the target quantity $p(s \rightsquigarrow t, D)$ is shown by the following result.

Lemma 1

$$p(s \rightsquigarrow t, D) = \sum_{T : s,t \in T} g_s(N, T) .$$

Proof. The definitions directly yield

$$p(s \rightsquigarrow t, D) = \sum_{L} \sum_{\substack{A \subseteq L \\ s \rightsquigarrow t \text{ in } A}} \prod_{v \in N} \rho_v(L_v)\beta_v(A_v) ,$$

the outer summation being over all linear orders L on N.

We next break the inner summation into two nested summations by observing that the sets $\mathcal{A}(L, N, T)$, for $s, t \in T$, form a partition of the set $\mathcal{A}(L) = \{A \subseteq L : s \rightsquigarrow t \text{ in } A\}$: indeed, each DAG $A \in \mathcal{A}(L)$ determines precisely one node set T such that A contains a path from s to v for exactly the nodes in $v \in T$. Thus we have

$$\begin{aligned} p(s \rightsquigarrow t, D) &= \sum_{L} \sum_{T : s,t \in T} \sum_{A \in \mathcal{A}(L,S,T)} \prod_{v \in N} \rho_v(L_v)\beta_v(A_v) \\ &= \sum_{T : s,t \in T} \sum_{L} \sum_{A \in \mathcal{A}(L,S,T)} \prod_{v \in N} \rho_v(L_v)\beta_v(A_v) \\ &= \sum_{T : s,t \in T} g_s(N, T) . \end{aligned}$$

This completes the proof. □

From the algorithmic point of view, the pair (S, T) is sufficient for enabling a factorization of the sum over the A_S into independent sums over the parent sets A_v, for $v \in S$. Indeed, we have the following recurrence.

Lemma 2

$$g_s(S,T) = 1 \qquad \textit{for } S = \emptyset \textit{ and } T = \emptyset\,,$$
$$g_s(S,T) = 0 \qquad \textit{for } s \notin T \textit{ and } (s \in S \textit{ or } T \neq \emptyset)\,,$$
$$g_s(S,T) = \sum_{v \in S} g_s(S \setminus \{v\}, T \setminus \{v\}) \rho_v(S \setminus \{v\}) \bar{\beta}_v(S,T) \qquad \textit{otherwise},$$

where

$$\bar{\beta}_v(S,T) = \begin{cases} \sum\limits_{\substack{A_v \subseteq S \setminus \{v\} \\ A_v \cap T \neq \emptyset}} \beta_v(A_v) & \textit{if } v \in T,\ v \neq s, \\ \sum\limits_{A_v \subseteq (S \setminus \{v\}) \setminus T} \beta_v(A_v) & \textit{if } v \in S \setminus T \textit{ or } v = s. \end{cases}$$

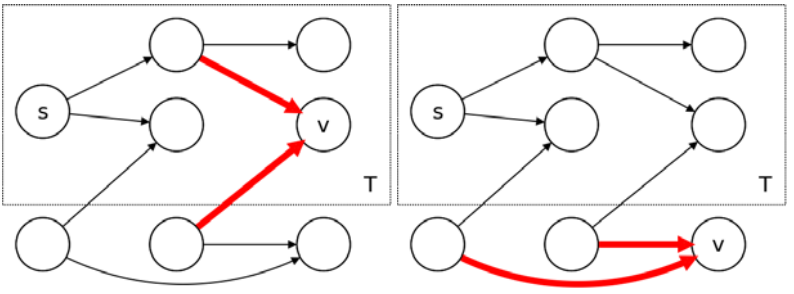
Proof. Proof is by straightforward induction on the size of S. First, observe that the sum over L_S in the definition of $g_s(S,T)$ breaks into a double-summation, in which the outer summation is over the last node $v \in S$ in the order L_S and the inner summation is over all linear orders, $L_{S\setminus\{v\}}$, on the remaining nodes $S \setminus \{v\}$. Second, observe that the summation over $A_S \in \mathcal{A}(L_S, S, T)$ breaks into a double-summation, in which the outer summation is over the DAGs $A_{S\setminus\{v\}} \in \mathcal{A}\big(L_{S\setminus\{v\}}, S \setminus \{v\}, T \setminus \{v\}\big)$ and the inner summation is over the parent sets $A_v \subseteq S \setminus \{v\}$ satisfying the requirement that (a) if there is no path from s to v (i.e., $v \notin T$), then there must be no path from s to u for any parent $u \in A_v$ of v, and (b) if there exists a path from s to v (i.e., $v \in T$), then there must exist a path from s to u for at least one parent u of v. □

Figure 1 illustrates the requirements on choosing parent sets for the node v in the last equation in Lemma 2. In Figure 1(a) $v \in T$, that is, it is required that there is a path from s to v. Now, we can choose any parent set for v as long as at least one of the parents is in T. On the other hand, in Figure 1(b) $v \notin T$, that is, it is required that there is no path from s to v. Now, we have to choose the parents of v from $S \setminus T$.

The evaluation of the values $g_s(S,T)$ using the recurrence is complicated by the fact that the inner summation, $\bar{\beta}_v(S,T)$, is over exponentially many sets A_v and, furthermore, there is a condition that depends not only on the set S but the set T. Fortunately, the inner summation can be precomputed for each $v \in N$ and $S \in N \setminus \{v\}$. Indeed, if $v \notin T$, then the sum is over all subsets A_v of $(S \setminus \{v\}) \setminus T$; if $v \in T$, then the sum is over all the remaining subsets of $S \setminus \{v\}$. Thus, it suffices to precompute

$$\hat{\beta}_v(U) = \sum_{A_v \subseteq U} \beta_v(A_v)$$

for all $U \subseteq N \setminus \{v\}$; the sums for the cases $v \notin T$ and $v \in T$ are then obtained as $\hat{\beta}_v((S \setminus \{v\}) \setminus T)$ and $\hat{\beta}_v(S \setminus \{v\}) - \hat{\beta}_v((S \setminus \{v\}) \setminus T)$, respectively. The function $\hat{\beta}_v$ is known as the zeta transform of β_v (over the subset lattice of $N \setminus \{v\}$), which can be



(a) (b)

Fig. 1. Choosing parent sets for a node $v \in S$ when (a) $v \in T$ and (b) $v \notin T$

computed, given β_v, by the so-called fast zeta transform algorithm (see, e.g., [13,15]) in time $O(n2^n)$ and space $O(2^n)$.

In summary, the values $g_s(S,T)$ for all $S \subseteq N$ and $T \subseteq S$ can be computed in time $O(n3^n)$ and space $O(3^n)$. The precomputation of the inner sum takes time $O(n^2 2^n)$ and space $O(n2^n)$ as noted above. Thus, the posterior probability that there exist a path from s to t, where s and t are two fixed nodes, can be computed in time $O(n3^n)$ and space $O(3^n)$. To compute the posterior probabilities for all node pairs st, it suffices to repeat the computations for each possible $s \in N$, for the values $g_s(S,T)$ actually contain the sufficient information regarding all possible descendant nodes t. Thus, in total, the time requirement is $O(n^2 3^n)$.

3 Experiments

Next we study how learning ancestor relations performs in practice. Our approach is to generate data from a known Bayesian network, called the *ground truth*, and compare the learned arcs and ancestor relations to the ground truth. Obviously, the learning performance is not expected to be perfect: when there are unobserved nodes at work, we easily learn arcs that are not present in the ground truth; this happens especially when an unobserved node is a common parent of two nodes that are not connected by an arc; namely, the two nodes are marginally dependent, and thus, in absence of the common parent, it is likely that we learn an arc between them, a false positive. On the other hand, we may expect that much of the structure can be learned even in presence of unobserved nodes. For example, if an unobserved node has exactly one child and one parent in the ground truth, both observed, then it is likely that the two arcs through the unobserved node in the middle will be just contracted to a single arc, which encodes a correct ancestor relation. We call the graph obtained from the ground truth by such contractions—that is, by connecting each parent of an unobserved node to every child of the node—the *shrunken ground truth*.

We have implemented the algorithm of Section 2.1 for Bayesian learning of ancestor relations in Matlab. In the experiments discussed next, we have used the BDeu score with the equivalent sample size of 1, a uniform prior over linear orders on the nodes, and a uniform prior over parent sets of size at most a user-defined bound, which we set to 6.

3.1 Challenges of Learning Ancestor Relations

It is instructive to examine some representative challenges we face when learning ancestor relations and arcs. We consider a Bayesian network whose DAG is shown in Figure 2(a). All 14 variables are binary. The parameters of the network, that is, the probability of a node taking the value 1 given a particular value combination of its parents was drawn uniformly at random from the range $[0, 1]$ for each node and value combination of its parents. We generated 10 000 samples from the Bayesian network and learned ancestor relations from the data. Note that there are 16 arcs and 39 ancestor–descendant pairs in the ground truth. The DAG has quite a large Markov equivalence class, 140 graphs in total, and so one cannot expect reliable deduction of ancestor relations from a single MAP DAG.

For clarity of presentation, we discuss our findings mainly in terms of arcs instead of ancestor relations. Figure 2(b) shows arcs that are assigned a posterior probability of 0.5 or larger. Suppose we claim every arc or ancestor relation with probability 0.5 or larger to be present. Then, in total there are 12 true positive arcs, 4 false positive arcs, 20 true positive ancestor relations, and 4 false positive ancestor relations. Inspection reveals that the ancestor relation errors are due to a few flipped arcs. For example, in the ground truth there is a path from node 1 to eight different nodes. Thus, flipping the arc from 1 to 2 causes one false positive and eight false negative ancestor relations. While arc errors are rather independent, one flipped arc can lead to numerous ancestor relation errors, as seen earlier. It should also be noted that arc flips that are prone to cause a larger number of ancestor relation errors are also more probable. Namely, an arc is easily flipped when it does not break or create any v-structure, which is typically the case when one of the nodes is a source node in the ground truth.

The presence of unobserved nodes leads to claiming arcs between nodes that are only marginally dependent. In Figure 2(c) we see a DAG constructed from the arcs with probability 0.5 or larger when nodes 1, 4, 7, and 11 are discarded. Node 1 does not have children, so its disappearance should not affect the structure among the rest of the nodes. However, the removal of nodes 4, 7, and 11 affects the rest of the nodes: For instance, node 11 is a common cause of nodes 13 and 14, and so an arc appears between nodes 13 and 14. Also, nodes 5 and 6, which are parents of node 7 in the ground truth, have become parents of node 10, a child of node 7 in the ground truth. Similarly, the removal of node 4 leads also to appearance of some direct arcs from its parents to its children. After discarding the unobserved nodes, the shrunken ground truth contains 14 arcs and 18 ancestor relations. The algorithm finds 8 true positive arcs, 6 false positive arcs, 11 true positive ancestor relations, and 7 false positive ancestor relations. This suggests that ancestor relations can sometimes be learned as well as individual arcs.

For comparison, we also learned a *partial ancestral graph* (PAG) from the data with unobserved nodes using the fast causal inference (FCI) algorithm [18], which is designed for causal discovery with unobserved variables. The output graph is shown in Figure 2(d). An arc marked with two arrowheads indicates that the algorithm claims the two nodes have a common (unobserved) cause; the symbol $\circ$ is a wildcard, indicating that there can be an arrowhead or there is no arrowhead. The results are generally in good agreement with the ground truth. The FCI algorithm is able to detect the unobserved parent of nodes 12 and 13. However, it is not sure whether there is an unobserved

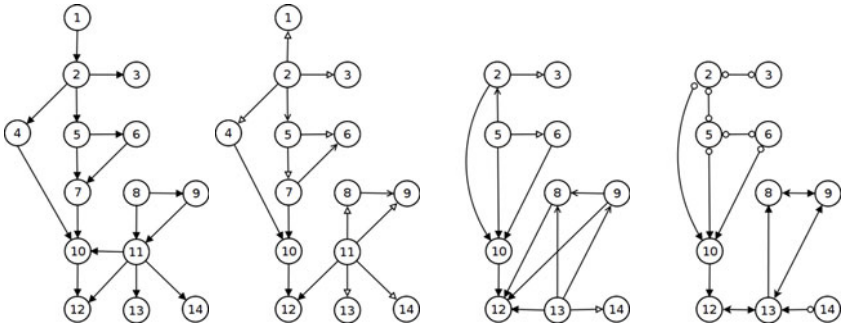


Fig. 2. Graphs. (a) The ground truth, from which 10 000 samples were generated. (b) Arcs with posterior probability at least 0.5. The arrowheads $>$, $\triangleright$, and $\blacktriangleright$ indicate that the probability is in the interval $(0.5, 0.8]$, $(0.8, 0.99]$, or $(0.99, 1]$, respectively. (c) Arcs with posterior probability at least 0.5 when nodes 1, 4, 7, and 11 are not observed. (d) A partially directed graph learned using the FCI algorithm when nodes 1, 4, 7, and 11 are not observed.

parent between nodes 13 and 14, and it is unable to detect the unobserved parent between nodes 12 and 14. It also finds an unobserved parent between nodes 8 and 9, which is not in agreement with the ground truth. As the wildcards assigned by the FCI algorithm do not quantify the uncertainty about the associated arcs, but the algorithm is ignorant regarding some ancestor relations, the algorithm may loose statistical power in detecting such relations; we will examine and discuss this issue further in the next section.

3.2 A Simulation Study

We generated synthetic data by a procedure adopted from Koivisto [14]. One hundred BNs on 14 binary nodes and maximum indegree 4, each with 10000 data points were obtained as follows.

1. Draw a linear order L on the node set $\{1, 2, \ldots, 14\}$ uniformly at random (u.a.r.).
2. For each node v independently:
   (a) let d_v be the number of predecessors of v in L;
   (b) draw the number of parents of v, denoted as n_v, from $\{0, 1, \ldots, \min\{4, d_v\}\}$ u.a.r.;
   (c) draw the n_v parents of v from the predecessors of v in L u.a.r.;
   (d) for each value configuration of the parents: draw the probability of a sample getting the value 1 from the uniform distribution on range $[0, 1]$.
3. Draw 10000 samples independently from the BN.

From each data set 24 subsets were generated by discarding $\ell = 0, 2, 4, 6, 8, 10$ randomly picked nodes and the associated data, and by including the first $m = 100, 500, 2000, 10000$ data points.

Our Bayesian method was applied to each data set and the performance of learning arcs and ancestor relations was summarized by ROC curves; see Figure 2. The ROC curve is obtained by setting a threshold for the posterior probability (of arcs or ancestor relations), and every time the posterior probability exceeds the threshold, we claim the respective arc or ancestor relation is present. Comparing these claims to the arc and ancestor relations that actually hold in the (shrunken) ground truth, we obtain true positives (TP) and false positives (FP) rates. By varying the threshold the pairs of these rates form a ROC curve, which shows the learning power (TP rate) as a function of FP rate.

As expected, the more data we have, the easier it is to learn both ancestor relations and arcs. Likewise, the task becomes harder as the number of unobserved nodes grows. (We note that the results for undirected arcs in the case of no unobserved nodes are in good agreement with Koivisto's [14] results for this particular setting.) The results (Figure 2) also suggest that the power of learning directed and undirected arcs is about the same, however, the power of learning ancestor relations being slightly smaller. The running times of the algorithm for 10, 12, and 14 observed nodes were roughly 3 minutes, 40 minutes, and 8 hours, respectively.

We then compared our Bayesian averaging approach to the deduction of structural features from a single MAP DAG. Two ways to pick a MAP DAG were considered: an optimistic and a random approach. In the optimistic approach we chose a member of the Markov equivalence class of a MAP DAG that yields the largest true positives rate, and used its true and false positives rates. This approach is arguably unrealistic in practice but serves as an upper bound for any approach based on a single MAP DAG. In the random approach we averaged the true and false positives rates over all DAGs in the Markov equivalence class of a MAP DAG; the averaged rates correspond to the respective expectations if one picks such a DAG at random. The true and false positives rates for these two approached are shown in Tables 1 and 2; column "diff." shows the difference between the true positives rates of the MAP DAG approach and the Bayesian averaging approach (the averages of the false positives rate being matched, of course); a negative value indicates that the Bayesian averaging approach is more powerful.

The results suggest that the random MAP DAG approach performs significantly worse than Bayesian averaging. On the other hand, the optimistic MAP DAG approach performs sometimes better than Bayesian averaging, especially when the data are abundant and there are many unobserved nodes.

Furthermore, we compared our method to the deduction of ancestor relations from the arc probabilities. To this end, we constructed a graph that consisted of the arcs whose posterior probability was larger than 0.5, that is, the arc is more likely to be present than absent, and deducted the ancestor relations from this graph. The results (Tables 1 and 2) show that the performance of the deduction of ancestor relations from arcs does not differ significantly from learning ancestor relations directly. We further compared the aforementioned approach to direct learning of ancestor relations. To this end, we assumed that exactly the ancestor relations whose probability is more than 0.5 exist and cross-tabulated the ancestor relations predictions for deducting the ancestor relations from arcs and the direct computation of ancestor relations; see Table 3. Table 3 shows the average number of the node pairs for which either both methods, only the deduction

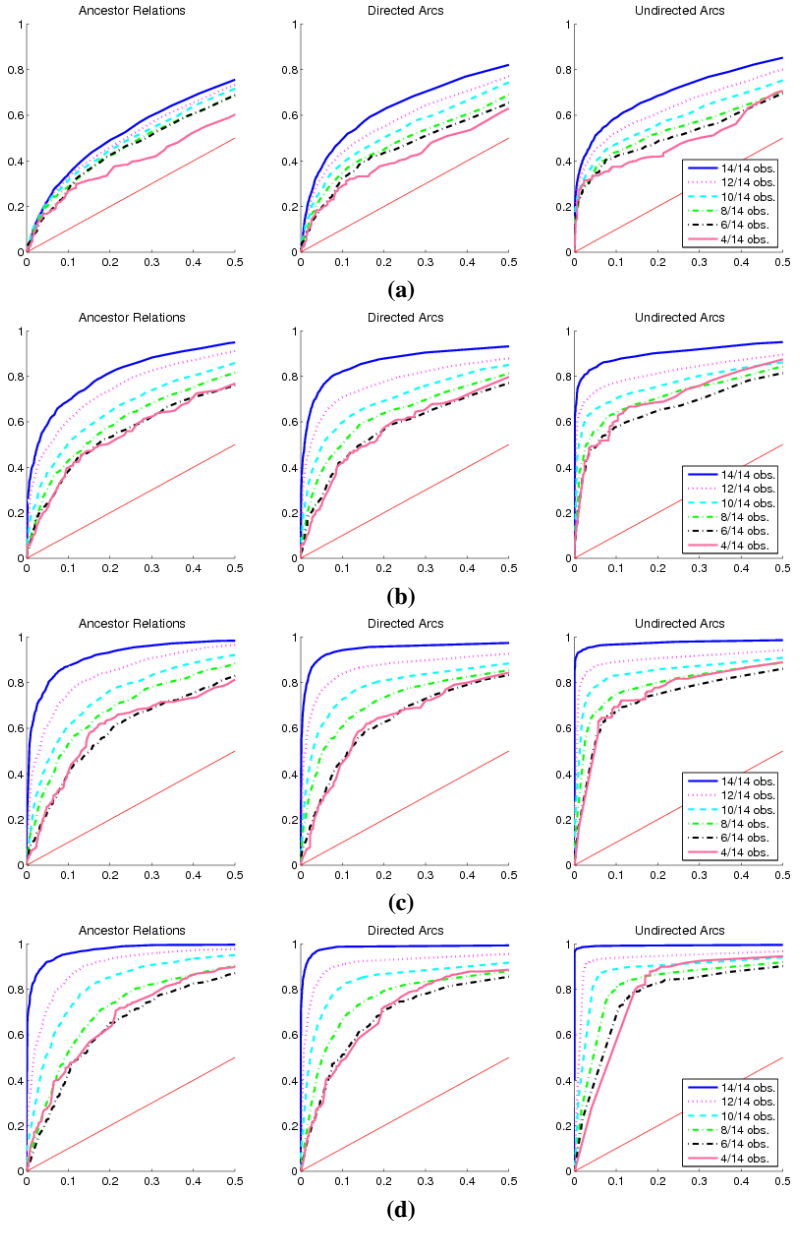


Fig. 3. ROC curves. The data contain (a) 100, (b) 500, (c) 2000 or (d) 10000 samples over 14 nodes. The straight red line is the curve obtained by random guess. The data-generating graphs contained on average 23.7 arcs and the shrunken ground truths on average 19.7, 15.9, 11.1, 7.0, and 3.3 arcs for 12, 10, 8, 6, and 4 observed nodes, respectively.

Table 1. Comparison of TP and FP rates for ancestor relations

		Opt. MAP DAG			Rand. MAP DAG			Arcs > 0.5			FCI		
m	ℓ	TP	FP	diff.	TP	FP	diff.	TP	FP	diff.	TP	FP	diff.
100	0	0.41	0.19	-0.09	0.37	0.21	-0.14	0.20	0.04	-0.01	0.002	0.000	0.002
100	2	0.35	0.16	-0.07	0.30	0.18	-0.14	0.17	0.03	-0.01	0.001	0.000	-0.001
100	4	0.34	0.11	0.01	0.27	0.14	-0.11	0.16	0.03	-0.01	0.002	0.000	0.002
100	6	0.34	0.08	0.07	0.24	0.12	-0.08	0.15	0.03	-0.00	0.002	0.000	0.002
100	8	0.36	0.05	0.19	0.23	0.09	-0.01	0.12	0.03	0.00	0.003	0.000	0.003
100	10	0.31	0.04	0.16	0.17	0.09	-0.06	0.12	0.02	0.01	0.000	0.000	0.000
500	0	0.65	0.10	-0.04	0.61	0.13	-0.12	0.58	0.04	0.01	0.014	0.002	-0.218
500	2	0.61	0.11	-0.02	0.54	0.14	-0.13	0.50	0.05	0.01	0.014	0.002	-0.143
500	4	0.53	0.11	0.01	0.45	0.14	-0.12	0.42	0.06	0.02	0.010	0.001	-0.073
500	6	0.50	0.10	0.06	0.40	0.14	-0.11	0.35	0.06	-0.01	0.011	0.000	-0.028
500	8	0.48	0.07	0.19	0.33	0.12	-0.10	0.27	0.06	0.00	0.014	0.000	0.014
500	10	0.51	0.05	0.26	0.32	0.12	-0.10	0.27	0.06	-0.02	0.005	0.000	0.005
2000	0	0.84	0.06	0.02	0.78	0.08	-0.08	0.78	0.05	0.01	0.048	0.004	-0.482
2000	2	0.76	0.10	0.02	0.70	0.12	-0.09	0.69	0.07	0.02	0.047	0.005	-0.329
2000	4	0.67	0.12	0.01	0.60	0.15	-0.11	0.60	0.09	0.02	0.041	0.007	-0.217
2000	6	0.64	0.12	0.06	0.54	0.16	-0.10	0.51	0.09	0.01	0.037	0.004	-0.090
2000	8	0.59	0.12	0.14	0.45	0.17	-0.09	0.40	0.10	-0.00	0.020	0.002	-0.033
2000	10	0.69	0.07	0.36	0.44	0.18	-0.18	0.41	0.09	0.07	0.005	0.000	0.005
10000	0	0.93	0.02	0.07	0.86	0.06	-0.06	0.87	0.02	0.00	0.129	0.011	-0.660
10000	2	0.86	0.08	0.06	0.79	0.11	-0.08	0.79	0.07	-0.00	0.121	0.010	-0.410
10000	4	0.80	0.11	0.07	0.70	0.15	-0.11	0.70	0.09	0.01	0.100	0.015	-0.326
10000	6	0.73	0.13	0.11	0.62	0.18	-0.10	0.60	0.13	0.00	0.086	0.010	-0.199
10000	8	0.72	0.14	0.19	0.57	0.20	-0.09	0.54	0.14	0.02	0.037	0.006	-0.125
10000	10	0.84	0.09	0.38	0.57	0.21	-0.11	0.54	0.14	-0.03	0.014	0.002	-0.025

from arc probabilities, only the direct computation of ancestor relation probabilities or neither method claims an ancestor relation to be present. Table 3 also shows the probability that the claim made by the direct computation is correct; NaN denotes that no claims falling into the particular category were made. Most of the time, both methods make the same predictions. Whenever the predictions differ, the prediction by direct computation is usually slightly more probable to be correct. We also notice that the two methods follow each other closely with larger datasets.

We also compared our method to the fast causal inference (FCI) method [18]; see Tables 1 and 2. We found it quite challenging to make a fair comparison because FCI outputs a partial ancestral graph (PAG) that cannot be directly compared to a DAG. We decided to ignore the wildcard arcs and claim only arcs and ancestor relations that FCI is sure about; this follows the approach of Spirtes et al. [19]. The results (Tables 1 and 2) show that FCI is very conservative: it does not make many mistakes but it often answers "don't know". This results in a relatively low statistical power of discovering arcs and ancestor relations, sometimes significantly lower than that of the Bayesian averaging approach (at matched FP rates).

One should notice, though, that FCI can discover unobserved nodes with some success. However, usually the unobserved nodes that FCI "finds," do not seem to match the

Table 2. Comparison of TP and FP rates for arcs

		Opt. MAP DAG			Rand. MAP DAG			Arcs > 0.5			FCI		
m	ℓ	TP	FP	diff.	TP	FP	diff.	TP	FP	diff.	TP	FP	diff.
100	0	0.34	0.05	-0.05	0.31	0.06	-0.10	0.26	0.03	0.00	0.003	0.000	0.003
100	2	0.30	0.06	-0.04	0.26	0.06	-0.11	0.22	0.02	0.00	0.002	0.000	-0.001
100	4	0.28	0.05	0.01	0.23	0.06	-0.08	0.18	0.02	0.00	0.003	0.000	0.003
100	6	0.28	0.04	0.05	0.21	0.06	-0.06	0.16	0.03	0.00	0.002	0.000	0.002
100	8	0.30	0.03	0.15	0.20	0.06	-0.02	0.13	0.03	0.00	0.003	0.000	0.003
100	10	0.31	0.03	0.15	0.18	0.07	-0.05	0.14	0.03	0.00	0.000	0.000	0.000
500	0	0.64	0.03	-0.03	0.60	0.03	-0.09	0.62	0.02	0.00	0.023	0.001	-0.273
500	2	0.55	0.03	0.00	0.50	0.04	-0.09	0.52	0.03	0.00	0.020	0.002	-0.153
500	4	0.46	0.04	0.02	0.41	0.05	-0.09	0.42	0.03	0.00	0.014	0.001	-0.076
500	6	0.42	0.04	0.06	0.34	0.06	-0.08	0.35	0.04	0.00	0.012	0.000	-0.026
500	8	0.42	0.04	0.18	0.30	0.07	-0.04	0.28	0.05	0.00	0.016	0.000	0.016
500	10	0.49	0.04	0.28	0.32	0.08	-0.08	0.30	0.07	0.00	0.005	0.000	0.005
2000	0	0.83	0.02	0.03	0.78	0.02	-0.06	0.81	0.02	0.00	0.075	0.003	-0.511
2000	2	0.71	0.03	0.02	0.66	0.04	-0.06	0.69	0.03	0.00	0.067	0.003	-0.319
2000	4	0.60	0.05	0.00	0.55	0.06	-0.08	0.59	0.04	0.00	0.054	0.005	-0.206
2000	6	0.55	0.05	0.05	0.47	0.07	-0.07	0.50	0.06	0.00	0.042	0.004	-0.088
2000	8	0.51	0.07	0.16	0.40	0.10	-0.05	0.41	0.08	0.00	0.023	0.002	-0.029
2000	10	0.65	0.06	0.33	0.43	0.12	-0.09	0.44	0.10	0.00	0.005	0.000	0.005
10000	0	0.93	0.01	0.08	0.87	0.02	-0.04	0.89	0.01	0.00	0.173	0.006	-0.672
10000	2	0.83	0.03	0.07	0.77	0.04	-0.05	0.80	0.03	0.00	0.146	0.006	-0.395
10000	4	0.75	0.05	0.08	0.67	0.06	-0.06	0.70	0.05	0.00	0.111	0.011	-0.304
10000	6	0.67	0.07	0.09	0.58	0.09	-0.06	0.61	0.08	0.00	0.090	0.009	-0.190
10000	8	0.64	0.09	0.15	0.52	0.12	-0.04	0.54	0.11	0.00	0.040	0.006	-0.118
10000	10	0.79	0.08	0.34	0.56	0.15	-0.09	0.58	0.14	0.00	0.015	0.002	-0.027

ones in the ground truth. For example, when the sample size is 2000 and there are no unobserved nodes, FCI finds on average 6.0 unobserved nodes. And when there are two unobserved nodes, only 11% of the "found" 4.8 unobserved nodes match the ground truth. In general, as the number of unobserved nodes increases, the number of found unobserved nodes decreases, but the percentage of correctly detected unobserved nodes increases; for example, when there are 8 unobserved nodes 54% of the claimed 0.7 unobserved nodes are correct.

3.3 Real Life Data

We tested our algorithm on two real life datasets found in UCI machine learning repository [7]: ADULT (15 variables, 32561 samples) and HOUSING (14 variables, 506 samples). We discretized all continuous variables to binary variables using the median as the cutpoint. Furthermore, we transformed the variable "native-country", which had 40 distinct values, of the ADULT dataset to a binary variable where 0 corresponded to value "USA" and 1 to all other values. For both datasets we set the maximum indegree to be 4.

Here, we do not know the ground truth and thus we have to resort to other comparisons. We investigate whether learning ancestor relations uncovers some information

Table 3. Ancestor Relations predicted by arcs and direct computation

		Predicted Ancestor Relations				Correct Predictions by dir. comp.			
m	ℓ	both	arcs	direct	none	both	arcs	direct	none
100	0	13.6	1.1	1.8	165.5	0.61	0.64	0.50	0.79
100	2	8.3	0.4	0.9	122.4	0.63	0.59	0.61	0.79
100	4	5.3	0.3	0.5	84.0	0.63	0.52	0.59	0.78
100	6	3.1	0.2	0.2	52.5	0.62	0.56	0.57	0.78
100	8	1.4	0.1	0.0	28.4	0.59	0.25	1.00	0.77
100	10	0.6	0.0	0.0	11.4	0.68	NaN	1.00	0.77
500	0	30.5	0.5	1.3	149.7	0.82	0.48	0.53	0.88
500	2	20.7	0.4	0.6	110.2	0.76	0.77	0.48	0.86
500	4	12.7	0.5	0.6	76.3	0.70	0.65	0.35	0.84
500	6	7.0	0.2	0.3	48.5	0.66	0.81	0.63	0.82
500	8	3.3	0.1	0.1	26.5	0.61	0.56	0.45	0.79
500	10	1.3	0.0	0.0	10.6	0.60	0.33	NaN	0.79
2000	0	39.7	0.2	0.4	141.8	0.85	0.82	0.39	0.93
2000	2	28.4	0.2	0.4	103.0	0.76	0.55	0.61	0.91
2000	4	18.6	0.2	0.4	70.8	0.69	0.47	0.30	0.88
2000	6	10.6	0.2	0.3	44.9	0.64	0.70	0.47	0.86
2000	8	5.2	0.1	0.1	24.6	0.58	0.89	0.54	0.82
2000	10	2.0	0.1	0.1	9.9	0.61	0.25	0.60	0.82
10000	0	40.9	0.1	0.4	140.7	0.92	0.60	0.61	0.96
10000	2	31.2	0.2	0.3	100.3	0.79	0.78	0.32	0.94
10000	4	21.6	0.1	0.2	68.0	0.71	0.60	0.36	0.91
10000	6	13.3	0.2	0.2	42.3	0.60	0.68	0.53	0.88
10000	8	7.2	0.0	0.1	22.7	0.57	0.75	0.44	0.85
10000	10	2.8	0.0	0.0	9.1	0.58	0.67	0.00	0.84

that cannot be obtained simply analyzing the arc probabilities. To this end, we deduct ancestor relations both from the ancestor relations probabilities and the arc probabilities. For ADULT, we have 210 potential ancestor relations. Both methods imply the presence of the same 79 ancestor relations. For HOUSING the methods are in almost as good agreement as for the ADULT. For 71 ordered pairs, both methods claim that an ancestor relation is present and for 110 pairs that an ancestor relation is not present. There is, however, one node pair for which the deduction from arcs suggests that there is no ancestor relation while the deduction from ancestor probabilities claims the opposite. This discrepancy is, though, due to the arbitrariness of the threshold. We notice that the posterior probability of an arc between the two nodes in question was 0.49 while the probability of an ancestor relation was 0.53.

4 Discussion

A key assumption in Bayesian network models is that all nodes relevant for capturing the dependencies of the associated variables are included in the model. One can argue that this assumption rarely holds in practice, and so the model is misspecified. Note, however, that in practice, every complex enough model is misspecified in one way or

another. The issue, in general, calls for robustness studies, disregarding whether the adopted statistical paradigm is a frequentist or a Bayesian one. In this paper we have studied the power of Bayesian structure discovery in Bayesian networks that do not explicitly model latent variables.

We contributed with two positive findings. First, we showed that Bayesian learning of ancestor relationships is computationally feasible when the number of observed nodes is moderate, say, fewer than 20. The algorithm resembles the dynamic programming algorithm of Koivisto and Sood [15] for computing the posterior probabilities of *modular* features, the main difference being in handling the *nonmodularity* of ancestor relations, which explains the somewhat larger computational complexity; this suggests that recent discussions of the limitation of the "exact Bayesian approach" to modular features [2,21] may be overly pessimistic. For larger networks on, say, more than 20 nodes, the dynamic programming algorithm becomes computational infeasible and one has to resort to heuristic methods, in particular, Markov chain Monte Carlo [6,9,10,16].

Second, our simulation study shows that ancestor relations can be discovered with reasonable power even when a large fraction of the nodes in the underlying data generating model are unobserved. For instance, with a sample of 10000 data points on 10 nodes, around 75 % of the ancestor relations that hold on the data generating network on 14 nodes are correctly detected at a false positive fraction of 12 %.

We also found that the presented Bayesian averaging approach outperforms some of its obvious rivals: the deduction of ancestor relations from a single MAP DAG and the popular constraint-based algorithm, FCI [19]. On the other hand, we found that full Bayesian averaging performs only marginally better than partial Bayesian averaging, that is, first inferring arcs based on their marginal posterior probabilities, with some fixed threshold, and then deducing ancestor relations from the so constructed DAG; this suggests that partial Bayesian averaging should be the method of choice when the number of nodes is about 20–30. Although someone may perceive the competitiveness of partial Bayesian averaging as a drawback for full Bayesian averaging, it should be noted that the insight about the competitiveness of partial Bayesian averaging was gained by being able to perform full Bayesian averaging. An intriguing open question we did not address this work is, how well some existing score-based heuristics [4] to discover unobserved nodes perform in terms of learning arcs and ancestor relations.

Acknowledgements. Authors like to thank anonymous reviewers for insightful comments and suggestions. This research was supported in part by the Academy of Finland, Grant 125637.

References

1. Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9(4), 309–347 (1992)
2. Eaton, D., Murphy, K.: Bayesian structure learning using dynamic programming and MCMC. In: Proceedings of the Twenty-Third Conference Annual Conference on Uncertainty in Artificial Intelligence, UAI (2007)
3. Elidan, G., Friedman, N.: Learning the dimensionality of hidden variables. In: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI), pp. 144–151 (2001)

4. Elidan, G., Friedman, N.: Learning hidden variable networks: The information bottleneck approach. Journal of Machine Learning Research 6, 81–127 (2005)
5. Elidan, G., Lotner, N., Friedman, N., Koller, D.: Discovering hidden variables: A structure-based approach. In: Advances in Neural Information Processing Systems (NIPS 2000), vol. 13 (2000)
6. Ellis, B., Wong, W.H.: Learning causal Bayesian network structures from data. Journal of American Statistical Association 103, 778–789 (2008)
7. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
8. Friedman, N.: Learning belief networks in the presence of missing values and hidden variables. In: Proceedings of the Fourteenth International Conference on Machine Learning, ICML (1997)
9. Friedman, N., Koller, D.: Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. Machine Learning 50(1-2), 95–125 (2003)
10. Grzegorczyk, M., Husmeier, D.: Improving the structure MCMC sampler for Bayesian networks by int roducing a new edge reversal move. Machine Learning 71, 265–305 (2008)
11. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20(3), 197–243 (1995)
12. Kang, E.Y., Shpitser, I., Eskin, E.: Respecting Markov equivalence in computing posterior probabilities of causal graphical features. In: Proceeding of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2010), pp. 1175–1180 (2010)
13. Kennes, R.: Computational aspects of the Möbius transformation of graphs. IEEE Transaction on Systems, Man, and Cybernetics 22(2), 201–223 (1992)
14. Koivisto, M.: Advances in exact Bayesian structure discovery in Bayesian networks. In: Proceedings of the Twenty-Second Annual Conference on Uncertainty in Artificial Intelligence, UAI (2006)
15. Koivisto, M., Sood, K.: Exact Bayesian structure discovery in Bayesian networks. Journal of Machine Learning Research 5, 549–573 (2004)
16. Madigan, D., York, J.: Bayesian graphical models for discrete data. International Statistical Review 63, 215–232 (1995)
17. Pearl, J.: Causality: Models, Reasoning, and Inference. Cambridge University Press, Cambridge (2000)
18. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. Springer, Heidelberg (2000)
19. Spirtes, P., Meek, C., Richardson, T.: Causal inference in the presence of latent variables and selection bias. In: Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence, UAI (1995)
20. Tian, J., He, R.: Computing posterior probabilities of structural features in Bayesian networks. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI (2009)
21. Tian, J., He, R., Ram, L.: Bayesian model averaging using the k-best Bayesian network structures. In: Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, UAI (2010)

ShareBoost: Boosting for Multi-view Learning with Performance Guarantees*

Jing Peng, Costin Barbu, Guna Seetharaman, Wei Fan, Xian Wu, and Kannappan Palaniappan

Montclair State University; MIT Lincoln Lab; AFRL/RITB; IBM T.J. Watson Research; IBM China Research; University of Missouri-Columbia
jing.peng@montclair.edu, barbu@ll.mit.edu, Gunasekaran.Seetharaman@rl.af.mil, weifan@us.ibm.com, wuxian@cn.ibm.com, palaniappank@missouri.edu

Abstract. Algorithms combining multi-view information are known to exponentially quicken classification, and have been applied to many fields. However, they lack the ability to mine most discriminant information sources (or data types) for making predictions. In this paper, we propose an algorithm based on boosting to address these problems. The proposed algorithm builds base classifiers independently from each data type (view) that provides a partial view about an object of interest. Different from AdaBoost, where each view has its own re-sampling weight, our algorithm uses a single re-sampling distribution for all views at each boosting round. This distribution is determined by the view whose training error is minimal. This shared sampling mechanism restricts noise to individual views, thereby reducing sensitivity to noise. Furthermore, in order to establish performance guarantees, we introduce a randomized version of the algorithm, where a winning view is chosen probabilistically. As a result, it can be cast within a multi-armed bandit framework, which allows us to show that with high probability the algorithm seeks out most discriminant views of data for making predictions. We provide experimental results that show its performance against noise and competing techniques.

Keywords: Data fusion, boosting, convergence, multi-view learning.

1 Introduction

Classifiers employed in real world scenarios must deal with various adversities such as noise in sensors, intra-class variations, and restricted degrees of freedom [18]. It is often helpful to develop classifiers that rely on data from various sources (views) for classification. Such classifiers require an effective way of fusing the

* A part of the research described in this material is based upon work funded by AFRL, under AFRL Contract No. FA8750-09-2-0155. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of MIT Lincoln Lab, or AFRL.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 597–612, 2011.

various sources of information. Resulting fused classifiers can offer a number of advantages, such as increased confidence in decision-making, resulting from fused complementary data, and robust performance against noise. Multi-view learning finds its applications in many domains such as defense, medicine, and sciences [13].

While algorithms that combine multi-view information are known to exponentially quicken object identification and classification, they lack the ability to seek out relevant information to augment a decision process. In this paper, we present a novel shared sampling approach to boosting for learning from multiple representations of data that addresses these problems. Here a representation (view) corresponds to a specific type of feature or attribute. For example, an image can be represented by (1) texture features, (2) edge features, and/or (3) shape features. Thus, each of these views provides a partial view about an object of interest, i.e., revealing a particular aspect of data. Our method provides a mechanism to exploit all of the data available, and as such, the method can be very useful for making inferences about potential objects of interest characterized with multiple views.

The proposed technique is a novel application of AdaBoost [10]. Similar to AdaBoost, our technique builds base classifiers independently from each view. Unlike AdaBoost, however, all views share the same sampling distribution as the view whose weighted training error is the minimum among all the views. This allows the most consistent data type to dominate over time, thereby significantly reducing sensitivity to noise. In addition, since the final strong classifier contains classifiers that are trained to focus on different views of the data, better generalization performance can be expected.

We note that each base classifier in the proposed algorithm is selected from one of the representations or views. We thus show that if we model base classifier selection as a sequential decision process, we can cast this scenario within a multi-armed bandit framework [3], where each view of data is modeled as the arm of a slot machine. The resulting algorithm can be viewed as a randomized version of the shared sampling algorithm in that a winning view is chosen probabilistically instead of in a greedy fashion. Furthermore, this casting allows us to show that with high probability the algorithm seeks out decision relevant views for making predictions. We also provide experimental results that corroborate our theoretical analysis.

2 Related Work

In multi-view learning, a co-training procedure for classification problems was developed [4]. The idea is that better classifiers can be learned at the individual view level, rather than constructed directly on all the available views. Co-training has been extensively investigated in the context of semi-supervised learning [22,23,8]. In this work, we are mainly interested in creating classifiers that fuse information from multiple views for better generalization.

In many ways, multi-view learning and data fusion address the same set of problems. From the viewpoint of data fusion, comprehensive surveys of various

classifier fusion studies and approaches can be found in [11,12]. More recently, Lanckriet et al. [13] introduce a kernel-based data fusion (multi-view learning) approach to protein function prediction in yeast. The method combines multiple kernel representations in an optimal fashion by formulating the problem as a convex optimization problem that can be solved using semi-definite programming.

In [24] stacked generalization from multiple views was proposed. It is a general technique for construction of multi-level learning systems. In the context of multi-view learning, it yields unbiased, full-size training sets for the trainable combiner. In some cases stacked generalization is equivalent to cross-validation, in other cases it is equivalent to forming a linear combination of the classification results of the constituent classifiers. In [25], a local learning technique was proposed that combines multi-view information for better classification.

Boosting has been investigated in multi-view learning recently [21]. In particular, there is a close relationship between our technique and that proposed in [21]. If we have a single view and base classifiers are allowed to include features as well, then both techniques reduce to AdaBoost. When noise exists, however, the two techniques diverge. The technique in [21] behaves exactly like AdaBoost. Noise forces the boosting algorithm to focus on noisy examples, thereby distorting the optimal decision boundary. On the other hand, our approach restricts noise to individual views, which has a similar effect to that of placing less mass of sampling probability on these noisy examples. This is the key difference between the two techniques.

Multi-armed bandits have been studied in a number of applications [2,3]. We state that multi-armed bandits described in [5] is stochastic by nature. There are many applications where the stochastic setting can be applied to nonstationary environments, such as performance tuning problems [15] and the SAT problem [14]. Algorithms such as UCB [2] and UCBV [1] work well for making AdaBoost more efficient. Given that AdaBoost is adversarial by nature, it is difficult to use stochastic bandits to derive strong performance guarantees on AdaBoost. Many arguments made in [5] remain heuristic to an extent. However, this has been addressed in [6].

3 Shared Sampling Algorithm

In this section, we first describe the shared sampling (ShareBoost) algorithm. We then present a randomized version of it. We are given a set of training examples: $X = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, and M disjoint features for each example $x_i = \{x_i^1, \cdots, x_i^M\}$, where $x_i^j \in \Re^{q_j}$, and $y_i \in \mathcal{Y} = \{-1, +1\}$. Each member x_i^j is known as a *view* of example x_i. We assume that examples (x_i, y_i) are drawn randomly and independently according to a fixed but unknown probability distribution D over $\mathcal{X} \times \mathcal{Y}$. Here the input space $\mathcal{X}$ is $\Re^q$, where $q = \sum_{j=1}^{M} q_j$.

The algorithm builds weak classifiers independently from each view (feature source). However, all data types share the same sampling distribution computed from the view having the smallest error rate. The key steps of the algorithm are shown in Algorithm 3, where $I(\cdot)$ is the indicator function.

ShareBoost $(\{(x_i^j, y_i)\}_{i=1}^n)$

1. **Initialization:** $w_1(i) = \frac{1}{n}$, $i = 1, \cdots, n$.
2. **For** $t = 1$ **to** T
   (a) Compute base classifier h_t^j using distribution w_t
   (b) Calculate:$\epsilon_t^j = \sum_i w_t(i) I(h_t^j(x_i^j) \neq y_i)$ and $\alpha_t^* = \frac{1}{2}\ln(\frac{1-\epsilon_t^*}{\epsilon_t^*})$, where $\epsilon_t^* = \min_j\{\epsilon_t^j\}$ with corresponding h_t^*
   (c) Update $w_{t+1}(i) = \frac{w_t(i)}{Z_t^*} \times \exp(-y_i h_t^*(x_i^*)\alpha_t^*)$, where Z_t^* is a normalization factor.
3. **Output:** $H(x) = sign(\sum_{t=1}^T \alpha_t^* h_t^*(x))$

Input to the algorithm is the jth view of n training examples. The algorithm produces as output a classifier that combines data from all the views. In the initialization step, all the views for a given training example are initialized with the same weight. The final decision function $H(x)$ is computed as a weighted sum of base classifiers $h_t^*(x^*)$, selected at each iteration from the views that had the smallest training error or largest α value. In this sense, ShareBoost possesses the ability to decide at each iteration which view to influence its final decision. This ability goes beyond simple subspace selection. It empowers ShareBoost not only to exploit the interplay between subspaces, but also to be more robust against noise.

We note that since we are mainly interested in asymtotatic margins, we are less concerned with distorted class probabilities associated with boosting predictions [9], especially in the two class case.

4 Randomized Shared Sampling Algorithm

The ShareBoost algorithm introduced above is greedy in that resampling weights for all views are determined solely by the winning view. That is, it employs a winner take-all strategy. One of the benefits associated with this algorithm is that noise will be restricted to individual views. In other words, noise will be *compartmentalized*, which has a similar effect to that of placing less mass of sampling probability on noisy examples. This, however, needs not to be the case in approaches such as those described in [21].

In order to provide a convergence analysis, we describe a randomized version of the shared sampling algorithm, where a winning view is chosen probabilistically. Consequently, it can be cast within a multi-armed bandit framework [3]. This in turn allows us to show that with high probability the algorithm chooses a set of *best* (large edges to be detailed later) views for making predictions.

4.1 Adversarial Multi-armed Bandit Approach

In the multi-armed bandit problem [17], a gambler chooses one of M slot machines to play. Formally, a player algorithm pulls one out of M arms at each time

t. Pulling an arm j_t at time t results in a reward $r_(j_t) \in [0,1]$, from a stationary distribution. The goal of the player algorithm is to maximize the expected sum of the rewards over the pulls. More precisely, let $G_A(T) = \sum_{t=1}^{T} r_(j_t)$ be the total reward that algorithm A receives over T pulls. Then the performance of algorithm A can be evaluated in terms of regret with respect to the average return of the optimal strategy (pulling consistently the best arm) $Reg = G_O - G_A(T)$, where $G_O = \sum_{t=1}^{T} \max_{i\in\{1,\cdots,M\}} R_(i_t)$. Here $R(i)$ represents the expected return of the ith arm.

In this work, we cast the proposed fusion algorithm within the adversarial bandit framework [2,3]. In this setup, no statistical assumptions are made about the generation of rewards. This can be viewed as having a second, non-random player that generates a reward sequence $\mathbf{r}_1, \mathbf{r}_2 \cdots$ of vectors

$$\mathbf{r}_t = (r_t(1), \cdots, r_t(M)),$$

where $r_t(j) \in [0,1]$. There is no restriction on reward vectors $\mathbf{r}(t)$ generated, which can be influenced by the player algorithm's previous actions. Only the reward $r_(j_t)$ of the chosen arm j_t is revealed to the player algorithm. Since the rewards are not drawn from a stationary distribution, any kind of regret can only be defined with respect to a particular sequence of actions. One such regret is the worst case regret $G_{(j_1,\cdots,j_T)} - G_A(T)$, where $G_{(j_1,\cdots,j_T)} = \sum_{j=1}^{T} r_t(j_t)$. Thus, the worst case regret measures how much the player algorithm lost (or gained) by following algorithm A instead of choosing actions $(i_i, \cdots, i_T)$.

A special case of this is the regret of strategy A for the best single action

$$Reg_A(T) = G_{max}(T) - G_A(T) \tag{1}$$

where $G_{max}(T) = \max_i \sum_{t=1}^{T} r_t(i)$. That is, strategy A is compared to the best fixed arm, retrospectively. Notice that when player algorithm A that achieves $\lim_{T\to\infty} \frac{Reg_A(T)}{T} \leq 0$ is called a no-regret algorithm.

4.2 Exp3.P: Exponential-Weight Algorithm for Exploration and Exploitation

The adversarial multi-armed bandit problem can be treated within the class of Exponentially Weighted Average Forecaster algorithms [7]. Typically these algorithms maintain a probability distribution over the arms and draws a random arm from this distribution at each step. The probability for pulling an arm increases exponentially with the average of past rewards the arm receives. In particular, we chose the Exp3.P (*E*xponential-weight algorithm for *E*xploration and *E*xploitation) algorithm [3], because the particular form of the probability bound on the weak regret (1) allows us to derive a strong result for the proposed fusion algorithm.

In the Exp3.P algorithm, the probability distribution (line 2(a)) for choosing arms is a mixture (weighted by γ) of the uniform distribution and a distribution that allocates a probability mass exponential in the estimated cumulative reward to each arm. This mixture ensures that the algorithm tries out all M arms. When

Exp3.P ($\alpha > 0$,$\gamma \in (0, 1]$)

1. **Initialization:** $i = 1, \cdots, M$.
   (a) $d_1(i) = exp(\frac{\alpha\gamma}{3}\sqrt{\frac{T}{M}})$
2. **For** $t = 1, 2, \cdots, T$
   (a) $p_t(i) = (1-\gamma)\frac{d_t(i)}{\sum_{j=1}^{M} d_t(j)} + \frac{\gamma}{M}$, $i = 1, \cdots, M$
   (b) Choose i_t randomly according to $p_t(i)$
   (c) Receive reward $r_t(i_t) \in [0, 1]$
   (d) For $j = 1, \cdots, M$ set

$$(1)\ \hat{r}_t(j) = \begin{cases} r_t(j)/p_t(j) & \text{if } j = i_t; \\ 0 & \text{otherwise.} \end{cases}$$

$$(2)\ d_{t+1}(j) = d_t(j)\exp(\frac{\gamma}{3M}(\hat{r}_t(j) + \frac{\alpha}{p_t(j)\sqrt{MT}}))$$

arm i_t is selected (line 2(d)(1)), the estimated reward $\hat{r}_(i_t)$ for the arm is set to $r_t(i)/p_t(i)$. This choice compensates the reward of arms that are unlikely to be chosen. For the purpose of our analysis, we state the following theorem (Theorem 6.3 in [3]).

Theorem 1. *For any fixed $T > 0$, for all $M \geq 2$ and for all $\delta > 0$, if $\gamma = \min\{3/5, 2\sqrt{(3M \log M)/(5T)}\}$ and $\alpha = 2\sqrt{\log(MT/\delta)}$, then*

$$G_{max} - G_{Exp3.P} \leq 4\sqrt{MT \log \frac{MT}{\delta}} + 4\sqrt{\frac{5}{3}MT \log M} + 8 \log \frac{MT}{\delta} \quad (2)$$

holds for any assignment of rewards with probability at least $1 - \delta$.

It can be seen that α and γ are "smoothing" parameters: the larger they are, the more uniform the probability distribution for choosing arms $\mathbf{p}_t$. In addition, Exp3.P is a no-regret algorithm with probability 1 [3].

4.3 Randomized ShareBoost: Combining ShareBoost and Exp3.P

We are now in a position to combine ShareBoost and Exp3.P to establish a performance guarantee for the proposed algorithm. To do so, we must first specify a reward function for each information source. First, we define the training error

$$Err = \frac{1}{n}|\{i : H(x_i) \neq y_i\}|. \quad (3)$$

If we write

$$E_H(H, W_1) = \sum_{i=1}^{n} w_1(i) exp(-H(x_i)y_i), \quad (4)$$

then $E_H(H, W_1)$ upper bounds Err [20]. Furthermore, let

$$E_h(h, W_t) = \sum_{i=1}^{n} w_t(i) exp(-h(x_i) y_i).$$

It can be shown that

$$E_H(H, W_1) = \prod_{t=1}^{T} E_h(h_t, W_t). \tag{5}$$

It is shown that at each boosting round, the base learner tries to find a weak classifier h_t that minimizes $E_h(h, W_t) = \sum_{i=1}^{n} w_t(i) exp(-h(x_i) y_i)$ (Algorithm 1). Thus, minimizing $E_h(h, W_t)$ at each boosting round minimizes the training error Eq. (3) in an iterative greedy fashion.

Now let

$$\beta_t = \sum_i w_i(t) y_i h_t(x_i) = E_{i \sim W(t)}[y_i h_t(x_i)], \tag{6}$$

be the *edge* [19] of the base hypothesis h_t chosen by the base learner at time step t. Here the edge helps define reward functions in the proposed algorithm.

One can show that [20]

$$E_h(h, W) = \sqrt{1 - \beta_t^2}. \tag{7}$$

This implies that the training error of the final classifier is at most $\prod_{t=1}^{T} \sqrt{1 - \beta_t^2}$. This upper bound suggests several possible reward functions. For example, we can define the reward function as

$$r_t(j) = 1 - \sqrt{1 - \beta_t^2(V_j)}, \tag{8}$$

where $\beta_t(V_j)$ is the edge (6) of the classifier chosen by the base learner from source V_j at the tth boosting round. Since $\beta_t^2(V_j) \in [0, 1]$, this reward is between 0 and 1.

It is important to notice that a reward function in logarithm is proposed in [6] that restricts values the edge (6) can take. In contrast, our reward function does not have such a restriction. Also, (8) allows us to establish a sharper bound than the one in [6], as we will see later.

The combined algorithm, called Randomized ShareBoost or rShareBoost, is shown in Algorithm 4. Here, w_t denotes the distribution for sampling examples that is shared by all sensors, while d_t represents the weight for determining the distribution for sampling views.

rShareBoost seems to have departed significantly from the ShareBoost sampling algorithm. In ShareBoost, boosting is executed in parallel by all the information sources. In contrast, rShareBoost performs boosting along the view chosen by the bandit algorithm only. From the viewpoint of computation, rShareBoost is much more efficient, i.e., it is a fraction $(1/M)$ of the time required for

rShareBoost ($\alpha > 0$, $\gamma \in (0, 1]$, $\{(x_i, y_i)\}_{i=1}^n$)

1. $w_1(i) = \frac{1}{n}$, $i = 1, \cdots, n$. $d_1(j) = exp(\frac{\alpha\gamma}{3}\sqrt{T/M})$, $j = 1, \cdots, M$.
2. **For** $t = 1$ **to** T
   (a) $p_t(j) = (1-\gamma)\frac{d_t(j)}{\sum_{k=1}^{M} d_t(k)} + \frac{\gamma}{M}$
   (b) Let j be the view chosen using p_t
   (c) Obtain base classifier h_t^j using distribution w_t.
   (d) Calculate: $\epsilon_t^j = \sum_i w_t(i) I(h_t^j(x_i^j) \neq y_i)$. and $r_t(j) \in [0, 1]$ (8).
   (e) For $k = 1, \cdots, M$ set
      i. $\hat{r}_t(k) = r_t(k)/p_t(k)$ $(k = j)$, and 0 $(k \neq j)$
      ii. $d_{t+1}(k) = d_t(k) e^{\frac{\gamma}{3M}(\hat{r}_t(k) + \frac{\alpha}{p_t(k)\sqrt{MT}})}$
   (f) Let $\alpha_t^* = \frac{1}{2}\ln(\frac{1-\epsilon_t^*}{\epsilon_t^*})$, where $\epsilon_t^* = \epsilon_t^j$, $h_t^* = h_t^j$
   (g) Update $w_{t+1}(i) = \frac{w_t(i)}{Z_t} \times \exp(-y_i h_t^*(x_i^*)\alpha_t^*)$, where Z_t is a normalization factor.
3. **Output:** $H(x) = sign(\sum_{t=1}^{T} \alpha_t^* h_t^*(x))$

ShareBoost. In addition, ShareBoost is greedy, while rShareBoost is not. That is, the probability distribution for sampling training examples for all views is determined solely by the winning source. In rShareBoost, however, the information source selected by Exp3.P may not be the winning view. This provides rShareBoost with an opportunity to examine potential information sources that may prove to be useful.

5 Convergence Analysis of Randomized ShareBoost

We prove a convergence result for the rShareBoost algorithm described in 4.3 in the following theorem. Since rShareBoost is a randomized version of ShareBoost, the result also provides insight into the behavior of the ShareBoost algorithm.

Theorem 2. *Let $V = \{V_1, \cdots, V_M\}$ be a set of M information sources. Suppose that there exists an information source $V_i \in V$ and a constant $0 < \rho <= 1$ such that for any distribution over the training data set S, the base learner returns a base classifier from V_{i^+} with an edge $\beta_{V_{i^+}} \geq \rho$. Then, with probability at least $1 - \delta$, the training error (3) of rShareBoost will become 0 after at most in time polynomial in (besides other parameters).*

$$(\log(M/\delta), 1/\rho, \log n),$$

where input parameters to rShareBoost are set to

$$\gamma = \min\{3/5, 2\sqrt{(3M \log M)/(5T)}\}$$

and

$$\alpha = 2\sqrt{\log(MT/\delta)},$$

as required by Theorem 3. More precisely,

$$T = \max\left(\log^2 \frac{M}{\delta}, \left(\frac{2C}{\rho - 2}\right)^4, \frac{2\log n}{\rho}\right). \tag{9}$$

where $C = \sqrt{32M} + \sqrt{27M \log M} + 16$.

Proof. The arguments are along the line of the proof of Theorem 1 in [6]. Once the number of sensor sources is given, M becomes constant. Let

$$r^* = \max_i \sum_{t=1}^{T} r_t(i) \tag{10}$$

be the reward of the optimal arm, retrospectively. We can denote this arm by $i^* = \arg\max_i \sum_{t=1}^{T} r_t(i)$. Thus r^* can be written as $r^* = \sum_{t=1}^{T} r_t(i^*)$. Notice that i^* may not be the same as arm i^+ returned by the weak learner. Since i^* is the best arm, it follows that

$$r^* \geq \sum_{t=1}^{T} r_t(i^+). \tag{11}$$

We first upper bound the logarithm of the exponential margin loss (4). From (4) and (5), we have

$$\log(E_H(H, W_1)) = \log\left(\prod_{t=1}^{T} E_h(h_t, W_t)\right) = \sum_{t=1}^{T} \log(E_h(h_t, W_t)).$$

¿From (7), (8) and $\log x \leq x - 1$, we can show that

$$\sum_{t=1}^{T} \log(E_h(h_t, W_t)) \leq \sum_{t=1}^{T} -r_t(j_t). \tag{12}$$

Let $Z = 4\sqrt{MT \log \frac{MT}{\delta}} + 4\sqrt{\frac{5}{3} MT \log M} + 8 \log \frac{MT}{\delta}$. From Theorem 1 and (10), we can bound (12) as follows $\sum_{t=1}^{T} -r_t(j_t) \leq -Tr^* + Z \leq -\sum_{t=1}^{T} r_t(i^+) + Z = \sum_{t=1}^{T} (\sqrt{1 - \beta_t^2(V_{j_t})} - 1) + Z$, where the second inequality follows from (11), and the equality follows from (8). We therefore have

$$\sum_{t=1}^{T} -r_t(j_t) \leq \sum_{t=1}^{T} (\sqrt{1 - \rho^2} - 1) + Z \tag{13}$$

where (13) follows from the assumption that $\beta_t(j^+) \geq \rho$. If we write

$$\sqrt{1 - \rho^2} = \sqrt{(1 - \rho)(1 + \rho)} = \sqrt{(1 - \rho) + (1 - \rho)\rho},$$

we can show that

$$\sqrt{1 - \rho^2} \leq 2 - \rho, \tag{14}$$

where we use the following facts: $\sqrt{x+y} \leq \sqrt{x}+\sqrt{y}$ for any $x, y \geq 0$, $\sqrt{(1-x)x} \leq \sqrt{1-x}$ for $0 \leq x \leq 1$, and $2xy \leq x^2+y^2$ for any x, y. Combining (13) and (14), we have

$$\sum_{t=1}^{T} \log(E_h(h_t, W_t)) \leq \sum_{t=1}^{T}(1-\rho) + Z. \tag{15}$$

Furthermore,

$$Z \leq 4\sqrt{MT(\log\frac{M}{\delta} + \sqrt{T})} + 4\sqrt{\frac{5}{3}MT\log M} + 8\log\frac{M}{\delta} + 8\sqrt{T} \tag{16}$$

$$\leq 4\sqrt{MT(\sqrt{T}+\sqrt{T})} + 4\sqrt{\frac{5}{3}MT\log M} + 8\sqrt{T} + 8\sqrt{T} \tag{17}$$

$$= T^{3/4}\sqrt{32M} + T^{1/2}\left(\sqrt{\frac{80}{3}M\log M} + 16\right) \tag{18}$$

$$\leq T^{3/4}\left(\sqrt{32M} + \sqrt{27M\log M} + 16\right), \tag{19}$$

where the first inequality follows from $\sqrt{T} > \log T$; the second inequality follows from $\sqrt{T} > \log T$ and $T > \log^2 \frac{M}{\delta}$; and the last inequality follows from the fact that $T > 1$ and $\frac{80}{3} < 27$. Now let $C = \sqrt{32M} + \sqrt{27M\log M} + 16$. Thus, $\sum_{t=1}^{T} \log(E_h(h_t, W_t)) \leq T(1-\rho+T^{-1/4}C) \leq -\frac{1}{2}T\rho$, where the last inequality follows from $T > (\frac{2C}{\rho-2})^4$ (9).

In [20], it is shown that $E_H(H, W_1)$ (4) upper bounds Err (3). That is $Err \leq E_H(H, W_1)$. Thus, we have $Err \leq \exp(-\frac{1}{2}T\rho)$. If we set Err to be less than $\frac{1}{n}$, it must be zero. Therefore, if we let

$$\exp(-\frac{1}{2}T\rho) \leq \frac{1}{n},$$

and combine with (9), we obtain the time bound stated in the theorem.

Notice that because of (8), our bound is $O(\frac{1}{\rho^4})$ in terms of ρ in the worst case, which is sharper than the time complexity of $O(\frac{1}{\rho^6})$ established in [6].

There are a number of reasons why the above bound makes sense. As noted in [6], the most significant is that the regret bound for Exp3.P does not depend on n. Another observation is that rShareBoost is a boosting algorithm. Thus, it should output a strong classifier H with zero training error after $T = O(\log n)$ iterations, when its base learner is capable of returning a base classifier having edge (6) $\beta_t \geq \rho$ for given $\rho > 0$. Several boosting algorithms meet this condition [20]. Notice that rShareBoost is PAC learnable because its time complexity is polynomial in $\log\frac{1}{\delta}$.

Both the number of information sources M and the quality that each source can provide in terms of ρ are involved in the analysis. When M is large, a large number of trials must be taken to gather information so that the best information source can be properly identified. On the other hand, when sensor sources are unable to provide reliable information, we must lower ρ, hence a reduced edge

Fig. 1. FERET Sample images

(recall large edge values imply large asymptotic margins [19]). Thus, a large number of iterations are required to build enough base classifiers to create a strong final classifier. In practice, a trade-off between the competing goals has to be made.

It is important to note that the theorem states that when views can provide useful information (a large edge, thus a large ρ), rShareBoost can quickly identify those views. That is, the probability of choosing those views to build base classifiers increases exponentially, which can be particularly useful in environments with noise, thereby avoiding unproductive computations. It addresses one of the important issues facing fusion: Can an algorithm automatically switch between subsets of information sources that are most decision-relevant? The above analysis shows that rShareBoost can.

6 Experiments

We have carried out empirical study evaluating the performance of the proposed algorithm. As comparison, the following methods are evaluated. (1) ShareBoost (Algorithm 1), and (2) rShareBoost. (Algorithm 4), (3) iBoost (The boosting with independent sampling distribution). In iBoost, re-sampling weights of training examples are independent for each view. Similar to ShareBoost, a base classifier from the view having the largest α value is selected at each boosting round. All these three algorithms employ a Naive Bayes learner to build base classifiers, and the Gaussian distribution is used for marginal's. The number of base classifiers is 150. (4) The semi-definite programming (SDP) algorithm [13], where kernel functions are Gaussian. (5) The AdaBoost-MV algorithm, where AdaBoost is applied to each view independently and final classification is determined by majority vote. (6) The AdaBoost-Ca algorithm, where AdaBoost is applied to a concatenation of all views. (7) The majority vote (SVM-MV) algorithm, where SVMs are used as component classifier. (8) The stacked generalization (Stacking) algorithm [24]. Stacking is very similar to SVM-MV. The only difference is that, instead of majority vote, the final combiner is another SVM.

Table 1. Average accuracy: Noise free

Data	*Face*	*Gender*	*Glass*	*Gene*
ShareBoost	0.76	0.87	0.74	0.67
rShareBoost	0.76	0.87	0.73	0.66
iBoost	0.75	0.85	0.72	0.60
SDP	0.70	0.45	0.48	0.60
AdaBoost-MV	0.74	0.77	0.71	0.62
AdaBoost-Ca	0.74	0.84	0.70	0.61
SVM-MV	0.70	0.58	0.58	0.64
Stacking	0.70	0.58	0.58	0.63

Table 2. Average accuracy: 30% noise

Data	*Face*	*Gender*	*Glass*	*Gene*
ShareBoost	0.72	0.63	0.63	0.56
rShareBoost	0.73	0.63	0.64	0.56
iBoost	0.65	0.53	0.56	0.54
SDP	0.70	0.46	0.52	0.54
AdaBoost-MV	0.68	0.51	0.58	0.53
AdaBoost-Ca	0.65	0.51	0.58	0.52
SVM-MV	0.69	0.57	0.59	0.53
Stacking	0.70	0.58	0.58	0.57

Three FERET image data sets and one gene data set are used here. The problems are (1) Face detection, (2) Gender classification, and (3) detection of Glasses on faces. Sample images are shown in Fig. 1.

For the face and gender data, each image is represented by three poses in terms of eigenfaces extracted from three head orientations: 1) frontal, 2) half left, and 3) half right profiles. The non-face images are blacked out faces. In the glass detection experiment, each image is represented by three types of features extracted from only one pose of an individual, namely (1) eigenfaces, (2) Canny edges, and (3) wavelet coefficients. Each dataset has 101 samples and each view has 101 dimensions after applying PCA.

The gene data set is from the Yeast Database (CYGD) [16]. The task is to combine different sources to determine membrane vs non-membrane proteins. Three sources are derived from BLAST and Smith-Waterman genomic methods, and from gene expression measurement. The dataset has 100 examples and the number of dimensions after applying PCA is 76, 74 and 64, respectively. These dimensions explain 90% variance in the data.

Ten-fold cross-validation was used for model selection. For rShareBoost, we set α to 0.15 and γ to 0.3, respectively, as suggested in [6]. The results are averaged over 30 runs (60% training and 40% testing). Table 1 shows the average accuracy (noise free). In terms of paired t–test with a 95% confidence level, for the Face data, ShareBoost and rShareBoost are significantly better than SDP, SVM-MV and Stacking. For the Gender data, ShareBoost and rShareBoost are better than

SDP, AdaBoost-MV, SVM-MV and Stacking. For the Glass data, ShareBoost and rShareBoost significantly outperform SDP, AdaBoost-Ca, SVM-MV and Stacking. For the Gene data, ShareBoost and rShareBoost outperform the rest, except SVM-MV and Stacking.

To examine sensitivity to noise, we randomly added 30% noise to the class label of the training data for all three views by *flipping* a label from one class to another. Flipping labels to generate noise produces similar effect to that produced by poor views in terms of overlapping classes. Table 2 shows the average classification accuracy over 30 runs in the noisy case. For *Face*, ShareBoost and rShareBoost ourperform iBoost, AdaBoost-MV, and AdaBoost-Ca. For Gender and Glass, ShareBoost and rShareBoost are significantly better than the rest. The results show that ShareBoost and rShareBoost are more than robust against noise than the competing methods. Overall, rShareBoost is similar to ShareBoost in performance. However, rShareBoost is much more efficient computationally than ShareBoost.

Note that we also carried out experiments using feature noise (white noise). Every method performed better. However, their relative performances remained the same as in the label noise case. Label noise seems to create harder problems. It most likely creates problems with overlapping classes, while feature noise (white noise) may not.

Notice that the number of base classifiers for both ShareBoost and rShare-Boost is 150. However, for each base classifier ShareBoost requires the amount

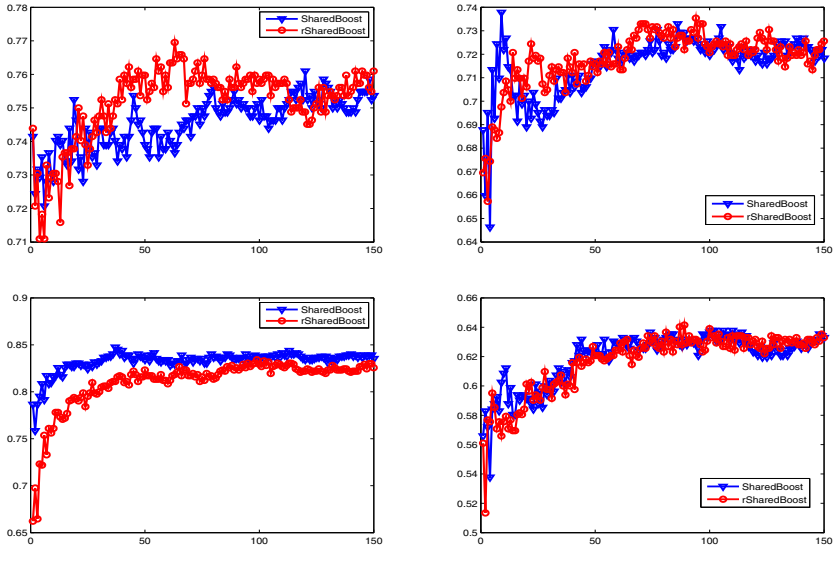


Fig. 2. Average accuracy as a function of the number of base classifiers achieved by ShareBoost and rShareBoost over 30 runs on the Face and Gender data. Left column: the noise-free case. Right column: the 30% noisy case.

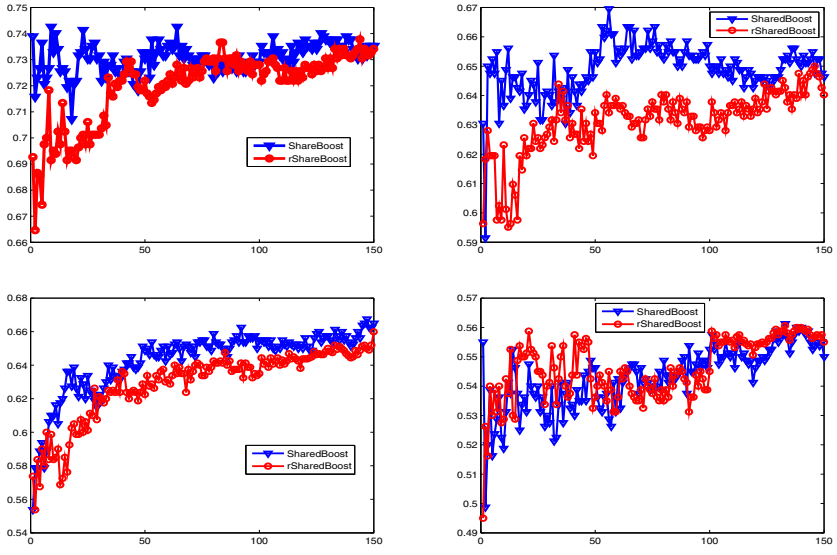


Fig. 3. Average accuracy as a function of the number of base classifiers achieved by ShareBoost and rShareBoost over 30 runs on the Glass and Gene data. Left column: the noise-free case. Right column: the 30% noisy case

of computation that is three times that required for rShareBoost. The fact that ShareBoost and rShareBoost registered similar performance on the problems examined shows that rShareBoost is more efficient computationally than ShareBoost. Figure 2 shows the average accuracy as a function of the number of base classifiers registered by ShareBoost and rShareBoost over 30 on the Face and Gender data, while Figure 3 shows the average accuracy by the two methods on the Glass and Gene data. In the figures, the left column shows the noise-free case, and the right column shows the 30% noisy case. The results show clearly that choosing a winning view probabilistically as in rShareBoost can be as effective as the winner-take-all strategy employed ShareBoost.

7 Summary

We have developed the ShareBoost algorithm for boosting for multi-view learning. The ShareBoost algorithm has been shown to be robust against noise by limiting noise to individual views. We have also developed a randomized version of the algorithm, rShareBoost, that can be cast within a multi-armed bandit framework. This formulation allows us to state its convergence and show that with high probability the rShareBoost algorithm judiciously seeks out decision relevant views for making predictions. rShareBoost has achieved a performance similar to ShareBoost at a fraction $(1/M)$ of time required for ShareBoost. We have provided the experimental results that validate our theoretical analysis.

We plan on exploring other base classifiers such as decision stumps in ShareBoost to examine its performance, and investigating its mechanism for exploiting interplays between multiple views for learning.

References

1. Audibert, J.-Y., Munos, R., Szepesvari, C.: Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. Theor. Comput. Sci. 410(19), 1876–1902 (2009)
2. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47, 235–256 (2002)
3. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.: The non-stochastic multi-armed bandit problem. SIAM Journal on Computing 32(1), 48–77 (2002)
4. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: In Proceedings of the Eleventh Annual Conference in Computational Learning Theory
5. Busa-Fekete, R., Kegl, B.: Accelerating adaboost using ucb. In: KDDCup (JMLR W&CP), pp. 111–122 (2009)
6. Busa-Fekete, R., Kegl, B.: Fast boosting using adversarial bandits. In: Proceedings of International Conference on Machine Learning (2010)
7. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
8. Culp, M., Michailidis, G., Johnson, K.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on Knowledge and Data Engineering 17, 1529–1541 (2005)
9. Fawcett, T., Niculescu-mizil, A.: Technical note: Pav and the roc convex hull. Machine Learning 68, 97–106 (2007)
10. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and Systems Science 55, 119–139 (1997)
11. Kittler, J.: Combining classifiers: A theoretical framework. Pattern Analysis and Applications 1, 18–27 (1998)
12. Kuncheva, L.I., Bezdek, J.C., Duin, R.P.W.: Decision templates for multiple classifier fusion: An experimental comparison. Pattern Recognition 34, 299–314 (2001)
13. Lanckriet, G.R.G., Deng, M.H., Cristianini, N., Jordan, M.I., Noble, W.S.: Kernel-based data fusion and its application to protein function prediction in yeast. In: Proceedings of the Pacific Symposium on Biocomputing, vol. 9, pp. 300–311 (2004)
14. Maturana, J., Fialho, A., Saubion, F., Schoenauer, M., Sebag, M.: Extreme compass and dynamic multi-armed bandits for adaptive operator selection. In: Proceedings of IEEE ICEC, pp. 365–372 (2009)
15. Mesmay, F.D., Rimmel, A., Voronenko, Y., Puschel, M.: Bandit-based optimization on graphs with application to library performance tuning. In: Proceedings of International Conference on Machine Learning, pp. 729–736 (2009)
16. Mewes, H.W., Frishman, D., Gruber, C., Geier, B., Haase, D., Kaps, A., Lemcke, K., Mannhaupt, G., Pfeiffer, F., Schüller, C., Stocker, S., Weil, B.: Mips: a database for genomes and protein sequences. Nucleic Acids Research 28, 37–40 (2000)
17. Robbins, H.: Some aspects of the sequential design of experiments. Bulletin American Mathematical Society 55, 527–535 (1952)

18. Ross, A., Jain, A.K.: Multimodal biometrics: an overview. In: Proceedings of 12th European Signal Processing Conference, pp. 1221–1224 (2004)
19. Rudin, C., Schapire, R., Daubechies, I.: Precise statements of convergence for adaboost and arc-gv. Contemporary Mathematics 443 (2007)
20. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence rated predictions. Machine Learning 3(37), 297–336 (1999)
21. Viola, P., Jones, M.: Fast and robust classification using asymmetric adaboost and a detector cascade. In: Advances in Neural Information Processing Systems, vol. 14 (2002)
22. Wang, W., Hua Zhou, Z.: On multi-view active learning and the combination with semi-supervised learning. In: Proceedings of the 25th International Conference on Machine Learning (2008)
23. Wang, W., Hua Zhou, Z.: A new analysis of co-training. In: Proceedings of the 25th International Conference on Machine Learning (2010)
24. Wolpert, D.H.: Stacked generalization. Neural Networks 5, 241–259 (1992)
25. Zhang, D., Wang, F., Zhang, C., Li, T.: Multi-view local learning. In: Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI), pp. 752–757 (2008)

Analyzing and Escaping Local Optima in Planning as Inference for Partially Observable Domains

Pascal Poupart[1], Tobias Lang[2], and Marc Toussaint[2]

[1] David R. Cheriton School of Computer Science
University of Waterloo, Ontario, Canada
ppoupart@cs.uwaterloo.ca
[2] Machine Learning and Robotics Lab
FU Berlin, Berlin, Germany
{tobias.lang,marc.toussaint}@fu-berlin.de

Abstract. Planning as inference recently emerged as a versatile approach to decision-theoretic planning and reinforcement learning for single and multi-agent systems in fully and partially observable domains with discrete and continuous variables. Since planning as inference essentially tackles a non-convex optimization problem when the states are partially observable, there is a need to develop techniques that can robustly escape local optima. We investigate the local optima of finite state controllers in single agent partially observable Markov decision processes (POMDPs) that are optimized by expectation maximization (EM). We show that EM converges to controllers that are optimal with respect to a one-step lookahead. To escape local optima, we propose two algorithms: the first one adds nodes to the controller to ensure optimality with respect to a multi-step lookahead, while the second one splits nodes in a greedy fashion to improve reward likelihood. The approaches are demonstrated empirically on benchmark problems.

1 Introduction

Toussaint et al. [20] recently showed that policy optimization in probabilistic domains is equivalent to maximizing the likelihood of normalized rewards. This connection between planning and inference has opened the door to the application of a wide range of machine learning and probabilistic inference techniques. However, policy optimization in partially observable domains is generally non-convex, including when reformulated as a likelihood maximization problem. As a result, policies often get stuck in local optima, which may be far from optimal.

In this paper, we analyze the local optima of finite state controllers in partially observable Markov decision processes (POMDPs) that are optimized by Expectation Maximization (EM). More precisely, we show that EM optimizes controllers by essentially performing a one-step lookahead where each parameter is adjusted in isolation (i.e., while keeping the other parameters fixed). We propose two techniques to help EM escape local optima. The first technique extends EM's one-step

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 613–628, 2011.

forward search to multiple steps. New nodes are added to the controller when the forward search detects a suboptimal action choice. The second approach splits controller nodes in two new nodes that are optimized by EM.

The paper is organized as follows. Sec. 2 reviews POMDPs, finite state controllers and planning as inference. Sec. 3 analyzes the properties of EM's local optima. Sec. 4 describes the two techniques to escape local optima. Sec. 5 evaluates the escape techniques on benchmark problems. Finally, Sec. 6 concludes the paper.

2 Background

2.1 Partially Observable Markov Decision Processes

Consider a partially observable Markov decision process (POMDP) described by a set $\mathcal{S}$ of states s, a set $\mathcal{A}$ of actions a, a set $\mathcal{O}$ of observations o, a stochastic transition function $\Pr(s'|s,a) = p_{s'|sa}$, a stochastic observation function $\Pr(o'|s,a) = p_{o'|sa}$ and a reward function $R(s,a) = r_{sa} \in \Re$. An important class of policies (denoted by π) are those representable by a stochastic finite state controller (FSC), which is a directed acyclic graph such that each node n chooses an action a stochastically according to an action distribution $\pi(a|n) = \pi_{a|n}$, each edge is labeled with an observation o' that chooses a successor node n' stochastically according to a successor node distribution $\Pr(n'|n,o') = \pi_{n'|no'}$ and the initial node is chosen stochastically according to $\Pr(n) = \pi_n$. The value $V(n,s) = V_{ns}$ of a FSC is the discounted sum of the rewards earned while executing the policy it encodes. We can compute this value by solving a linear system:

$$V_{ns} = \sum_a \pi_{a|n} r_{sa} + \gamma \sum_{s'o'n'} p_{s'|sa} p_{o'|sa} \pi_{n'|no'} V_{n's'} \ \forall ns$$

Hence, for a given initial state distribution (a.k.a. belief) $\Pr(s) = p_s$, the value of the policy is $\sum_{sn} p_s \pi_n V_{ns}$. Initial algorithms to optimize FSCs were based on policy iteration [6], however the size of the controller tends to grow exponentially with the number of iterations. Alternatively, since there are often good policies that can be represented by small controllers, one can search for the best controller with a fixed number of nodes. This search can be formulated as a non-convex optimization problem [1][1]:

$$\max_{\pi_n, \pi_{a|n}, \pi_{n'|no'}, V_{ns}} \quad \sum_{sn} p_s \pi_n V_{ns}$$

$$\text{s.t. } V_{ns} = \sum_{as'o'n'} \pi_{a|n} [r_{sa} + \gamma p_{s'|sa} p_{o'|sa} \pi_{n'|no'} V_{n's'}] \ \forall ns$$

[1] The optimization problem described in [1] assumes a fixed initial node and merges $\pi_{a|n}$ and $\pi_{n'|no'}$ into a single distribution $\pi_{an'|no'}$, but these differences are irrelevant for this paper.

$$\sum_a \pi_{a|n} = 1 \; \forall n, \; \pi_{a|n} \geq 0 \; \forall an$$
$$\sum_{n'} \pi_{n'|no'} = 1 \; \forall no', \; \pi_{n'|no'} \geq 0 \; \forall n'no'$$

Algorithms to find an optimal policy include gradient ascent [11], sequential quadratic programming [1], bounded policy iteration [14] and stochastic local search [3]. However, the non-convex nature of the problem generally prevents any of these algorithms from finding an optimal controller with certainty.

2.2 Planning as Inference

In another line of research, Toussaint et al. [19] proposed to reformulate policy optimization as a likelihood maximization problem. The idea is to treat rewards as random variables by normalizing them. Let $\bar{R}$ be a binary variable such that

$$\Pr(\bar{R}{=}true|s,a) = p_{\bar{r}_{true}|sa} = (r_{sa} - \min_{sa} r_{sa})/(\max_{sa} r_{sa} - \min_{sa} r_{sa}))$$

Similarly, we treat the decision variables A and N as random variables with conditional distributions corresponding to $\pi_{a|n}$, π_n and $\pi_{n'|no'}$. This gives rise to a graphical model where the value of a policy can be recovered by estimating the probability that each reward variable is true. However, rewards are discounted and added together, so Toussaint et al. propose to use a mixture of dynamic Bayesian networks (DBNs) where each DBN is t time steps long with a single reward variable at the end and is weighted by a term proportional to γ^t. Hence, the value of a policy is proportional to $\Pr(\bar{R}{=}true)$ in this mixture of DBNs. To optimize the policy, it suffices to search for the distributions π_n, $\pi_{n'|no'}$ and $\pi_{a|n}$ that maximize $\Pr(\bar{R}{=}true)$. This is essentially a maximum a posteriori estimation problem that can be tackled with algorithms such as Expectation Maximization (EM) [5]. More specifically, EM alternates between computing the expectations

$$E(n|\bar{R}{=}true, \pi^i) = E_{n|\bar{r}_{true}\pi^i}$$
$$E(a,n|\bar{R}{=}true, \pi^i) = E_{an|\bar{r}_{true}\pi^i}$$
$$E(n',o',n|\bar{R}{=}true, \pi^i) = E_{n'o'n|\bar{r}_{true}\pi^i}$$

and updating the distributions

$$\pi_n^{i+1} = E_{n|\bar{r}_{true}\pi^i} / \sum_n E_{n|\bar{r}_{true}\pi^i}$$
$$\pi_{a|n}^{i+1} = E_{an|\bar{r}_{true}\pi^i} / \sum_a E_{an|\bar{r}_{true}\pi^i}$$
$$\pi_{n'|o'n}^{i+1} = E_{n'o'n|\bar{r}_{true}\pi^i} / \sum_{n'} E_{n'o'n|\bar{r}_{true}\pi^i}$$

The expectations are obtained as follows

$$E_{n|\bar{r}_{true}\pi^i} = \sum_s p_s \pi^i_n \beta_{ns} \tag{1}$$

$$E_{an|\bar{r}_{true}\pi^i} = \sum_{ss'o'n'} \alpha_{sn} \pi^i_{a|n} [p_{\bar{r}_{true}|sa} + \gamma p_{s'|sa} p_{o'|s'a} \pi^i_{n'|o'n} \beta_{n's'}] \tag{2}$$

$$E_{n'o'n|\bar{r}_{true}\pi^i} = \sum_{ss'a} \alpha_{sn} \pi^i_{a|n} p_{s'|sa} p_{o'|s'a} \pi^i_{n'|o'n} \beta_{n's'} \tag{3}$$

where $\alpha = \lim_{t\to\infty} \alpha^t$ and $\beta = \lim_{t\to\infty} \beta^t$ are the forward and backward terms obtained in the limit according to the following recursions[2]:

$$\alpha^0_{sn} = p_s \pi_n$$
$$\alpha^t_{s'n'} = b_{s'} \pi_{n'} + \gamma \sum_{asno'} \alpha^{t-1}_{sn} \pi_{a|n} p_{s'|sa} p_{o'|as'} \pi_{n'|no'} \ \forall t > 0 \tag{4}$$

$$\beta^0_{sn} = \sum_a \pi_{a|n} p_{\bar{r}_{true}|sa}$$
$$\beta^t_{sn} = \sum_{as'n'o'} \pi_{a|n} [p_{\bar{r}_{true}|sa} + \gamma p_{s'|sa} p_{o'|as'} \pi_{n'|no'} \beta^{t-1}_{s'n'}] \ \forall t > 0 \tag{5}$$

An implication of the reformulation of policy optimization as an inference problem is that it opens the door to a variety of inference techniques and allows continuous [7], hierarchical [18], reinforcement learning [21] and multi-agent [8] variants to be tackled with the same machinery. Nevertheless, an important problem remains: policy optimization is inherently non-convex and therefore the DBN mixture reformulation does not get rid of local optima issues.

2.3 State Splitting

Siddiqi et al. [15] recently proposed an approach to discover the number of hidden states in HMMs by state splitting. Since there is no restriction on the number of hidden states, this approach can be viewed as a technique to escape local optima. In Section 4.2, we adapt this approach to POMDP controllers where internal nodes are split to escape local optima.

State splitting in HMMs works as follows. Run EM to learn the parameters of an HMM and Viterbi to find the most likely state paths. Then, for each state s, investigate the possibility of splitting s in two new states s_1 and s_2. Let T_s be the set of time steps where s is the most likely state. Replace the parameters that involve s by new parameters that involve s_1 or s_2. Optimize the new parameters with respect to the time steps in T_s. In other words, clamp the states outside of T_s to their most likely value and re-run EM to learn the parameters that involve s_1 and s_2 while keeping all other parameters fixed. At each iteration, greedily select the split that improves the model the most, then run full EM to converge to a new local optimum.

[2] In practice, $\alpha \approx \alpha^t$ and $\beta \approx \beta^t$ for large enough t depending on the discount γ.

3 Local Optima Analysis

When EM gets trapped in a local optimum, a simple strategy to escape consists of adding new nodes to the controller. However, unless the new nodes are carefully initialized, they will not help EM escape. For instance, as discussed in the experiments, adding random nodes is ineffective. Hence, there is a need to understand the conditions under which EM gets trapped so that the new nodes can be initialized to break those conditions. In this section, we show that EM stops making progress when the parameters of the nodes are optimal with respect to a one-step look ahead from a special set of beliefs. Based on this insight, in the next section, we propose an escape technique that adds nodes to the controller according to a multi-step lookahead.

Let's have a closer look at the policy updates performed by EM. In Eq. 1, 2 and 3 the policy terms π^i_n, $\pi^i_{a|n}$ and $\pi^i_{n'|o'n}$ can be factored out of the sum. This means that EM performs a multiplicative update:

$$\pi^{i+1}_n \propto \pi^i_n f_n \text{ where } f_n = \sum_s p_s \beta_{sn} \tag{6}$$

$$\pi^{i+1}_{a|n} \propto \pi^i_{a|n} g_{an} \text{ where } g_{an} = \sum_{ss'o'n} \alpha_{sn}[p_{\bar{r}_{true}|sa} + \gamma p_{s'|sa} p_{o'|s'a} \pi^i_{n'|o'n} \beta_{n's'}] \tag{7}$$

$$\pi^{i+1}_{n'|o'n} \propto \pi^i_{n'|o'n} h_{n'o'n} \text{ where } h_{n'o'n} = \sum_{ss'a} \alpha_{sn} \pi^i_{a|n} p_{s'|s,a} p_{o'|s'a} \beta_{n's'} \tag{8}$$

The multiplicative nature of the updates tells us that EM converges to a policy that chooses an initial node with maximal f_n, actions with maximal g_{an}, and successor nodes n' with maximal $h_{n'o'n}$. This is formalized by the following theorem.

Theorem 1. *If a policy π is a stable fixed point of EM (i.e., the fixed point is an attractor within an ϵ-hyperball), then*

$$\forall n \text{ if } \pi_n \neq 0 \text{ then } f_n = \max_n f_n \tag{9}$$

$$\forall an \text{ if } \pi_{a|n} \neq 0 \text{ then } g_{an} = \max_a g_{an} \tag{10}$$

$$\forall n'o'n \text{ if } \pi_{n'|o'n} \neq 0 \text{ then } g_{n'o'n} = \max_{n'} g_{n'o'n} \tag{11}$$

Proof. Suppose that π is a stable fixed point of EM. From (7) it follows that $\pi_{a|n} = c_n \pi_{a|n} g_{an}$ with a normalization constant c_n. For all non-zero $\pi_{a|n} \neq 0$, all actions in n have the same g-value $g_{an} = 1/c_n$. It remains to show that $1/c_n$ is also the maximum g_{an}, which we prove by contradiction. Let a^* be an action with $\pi_{a^*|n} = 0$ at the fixed point, but $g_{a^*n} > 1/c_n$. Consider an infinitesimal perturbation of the policy such that $\pi_{a^*|n} = \epsilon$. Independent of how small ϵ is, the probability of a^* will increase in the next iteration and $\pi_{a^*|n}$ will diverge from 0. Therefore, this fixed point is instable. The same reasoning holds for π_n with f_n and $\pi_{n'|o'n}$ with $h_{n'o'n}$.

Note that in practice, ensuring that parameters never become identically zero (e.g., by adding the smallest amount of noise in this case) ensures that EM always converges to such a stable fixed point.

The above theorem also gives a theoretical justification for the "smoothed greedy" update heuristics often used in practice [18]. Since EM will converge to a policy where $\pi_{a|n} = 0$ for all a, n where $g_{an} < \max_a g_{an}$, then it is reasonable to update $\pi_{a|n}$ by moving it towards a greedy policy that assigns non-zero probability only to the a, n pairs that are maximal in g (and similarly for π_n and $\pi_{n'|no'}$). Note however that there is no guarantee that such greedy updates will converge (they may lead to cycling), but they often speed up convergence in practice.

Let's interpret the conditions under which EM converges (i.e., conditions formalized in Theorem 1) since this will help us derive an escape technique in the next section. First, we point out that the backward terms β_{sn} in Eq. 5 can be interpreted as the total expected discounted rewards that will be earned when executing π from s, n. In other words, it is the value function V_{sn} of π at s, n. Similarly, the forward terms α_{sn} in Eq. 4 can be interpreted as the discounted occupancy frequency of π. In other words, α_{sn} indicates the expected number of times (discounted by γ^t for t time steps) that state s is reached in node n when executing π. We also define $b_{s|n} = \alpha_{ns} / \sum_s \alpha_{ns}$ to be the belief (e.g., state distribution) with respect to node n proportional to α_{ns}. Since $\alpha_{n\cdot}$ is the discounted sum of all reachable beliefs in node n, $b_{s|n}$ can be interpreted as a weighted average of the reachable beliefs in node n. Based on this, we can show that f_n is the expected value of π when starting its execution in node n. Similarly, g_{an} is proportional to the Q-function for belief $b_{s|n}$ and $h_{n'o'n}$ is proportional to the expected value of each successor node for the belief reachable from $b_{s|n}$ when executing π and receiving observation o'. This becomes evident when we replace α_{sn} by $b_{s|n}$ and β_{sn} by V_{sn} in Eq. 6, 7 and 8.

$$f_n \propto \sum_s p_s V_{ns} \tag{12}$$

$$g_{an} \propto \sum_s b_{s|n} [p_{\bar{r}_{true}|sn} + \sum_{s'o'n'} p_{s'|sa} p_{o'|s'a} \pi_{n'|o'n} V_{n's'}] \tag{13}$$

$$h_{n'o'n} \propto \sum_{sao'} b_{s|n} \pi_{a|n} p_{s'|sa} p_{o'|s'a} V_{n's'} \tag{14}$$

In the limit, since EM chooses an initial node with maximal f value (as well as actions with maximal g value and successor nodes with maximal h value), EM implicitly maximizes the initial node value for a fixed policy (as well as the Q-function for fixed successor node distributions and the successor nodes value for fixed action distributions). In the following, let us discuss how EM's convergence conditions are related to Bellman's global optimality conditions for controllers. More specifically, a controller is (globally) optimal when it satisfies the following condition for all beliefs b reachable in each node n:

$$V_{nb} = \max_a r_{ba} + \gamma \sum_{o'} p_{o'|ba} max_{n'} V_{n'b^{ao'}} \tag{15}$$

where $V_{nb} = \sum_s b_s V_{ns}$, $p_{o'|ba} = \sum_{ss'} b_s p_{s'|sa} p_{o'|s'a}$, $r_{ba} = \sum_s b_s r_{sa}$ and $b^{ao'}_{s'} \propto \sum_s b_s p_{s'|sa} p_{o'|s'a}$. The above equation can be rewritten in three optimality conditions for the choice of successor nodes, actions and initial node:

$$n^{bao'} = \operatorname*{argmax}_{n'} \sum_{ss'} b_s p_{s'|sa} p_{o'|s'a} V_{n's'} \ \forall bao' \tag{16}$$

$$a^n = \operatorname*{argmax}_{a} \sum_{s} b_s [r_{sa} + \gamma \sum_{s'o'} p_{s'|sa} p_{o'|s'a} V_{n^{bao's'}}] \ \forall n \tag{17}$$

$$n^0 = \operatorname*{argmax}_{n} \sum_{s} p_s V_{ns} \tag{18}$$

Note the similarity between these equations and the definitions of f, g and h in Eq. 12, 13 and 14. One important difference is that n^0, a^n and $n^{bao'}$ must be optimal for all beliefs b reachable in each node n to ensure global optimality, where as EM only ensures that the initial node, action and successor node choices are optimal for the single belief $b_{s|n}$ associated with each node n. Another important difference is that n^0, a^n and $n^{bao'}$ must be optimized simultaneously to ensure global optimality, where as EM adjusts the initial node, action and successor node distributions separately while keeping the other distributions fixed.

Below, in Theorem 2, we show that the convergence conditions described by Eq. 9, 10 and 11) are necessary, but not sufficient to ensure global optimality. Note that EM is already known to converge to local optima and therefore there must exist conditions that are necessary, but not sufficient. So the point of the theorem is to show that the particular conditions described by Eq. 9, 10 and 11) are indeed the ones for controller optimization. We encourage the reader to pay special attention to the proof since it shows that optimizing the parameters of the controller by a one-step lookahead from the beliefs $b_{s|n}$ associated with each node is a sensible thing to do even when these beliefs are not reachable. In the next section, we use this idea to develop a multi-step forward search from each $b_{s|n}$ instead of the initial belief to reduce the complexity of the search.

Theorem 2. *The conditions described by Eq. 9, 10 and 11 are necessary, but not sufficient, to ensure global optimality of FSCs.*

Proof. If $b_{s|n}$ is a reachable belief for each node n, then it is clear that the conditions described by Eq. 9, 10 and 11 are a subset of the global optimality conditions and therefore they are necessary, but not sufficient. However, the beliefs $b_{s|n}$ associated with each node n are not always reachable (even though they are weighted averages of the reachable beliefs). Nevertheless, we can still show that the conditions are necessary for optimality by observing that the beliefs $b_{s|n}$ are convex combinations of the reachable beliefs and therefore have the same optimal action and successor nodes as the reachable beliefs. Suppose that a controller is globally optimal, then $\pi_{a|n}$ and $\pi_{n'|o'n}$ must be optimal for all reachable beliefs of node n. We also know that $\pi_{a|n}$ and $\pi_{n'|o'n}$ are optimal for the convex hull of the reachable beliefs in node n since the optimal value function

Algorithm 1. Forward Search

Function: forwardSearch
Inputs: α, β, π and $timeLimit$
Output: $new\mathcal{N}$ (set of new nodes)
Let $b_{s|n} \propto \alpha_{ns}$ for each n
$d \leftarrow 0$
while $time < timeLimit$ **do**
  $d \leftarrow d + 1$
  for $n \in \mathcal{N}$ **do**
    $[new\mathcal{N}, gain] \leftarrow$
     $recursiveSearch(b_{s|n}, \beta, \pi, d)$
    if $gain > 0$ **then**
      return;
    end if
  end for
end while

Function: recursiveSearch
Inputs: b, β, π and d (depth)
Output: $new\mathcal{N}$ (new nodes) and $bestGain$
$bestGain \leftarrow 0$
$new\mathcal{N} \leftarrow \emptyset$
if $d = 1$ **then**
  Current value: $v \leftarrow \max_n \sum_s b_s \beta_{sn}$
  Compute v^*, $n^{ao'}$, a^* (Eq. 15-17)
  if $v^* - v > 0$ **then**
    $bestGain \leftarrow v^* - v$
    $new\mathcal{N} \leftarrow \{n\}$
      where $\pi_{a^*|n} = 1$, $\pi_{n^{a^*o'}|o'n} = 1$
  end if
else
  for $a \in A, o' \in O$ **do**
    $[\mathcal{N}, gain] \leftarrow$
     $recursiveSearch(b^{ao'}, \beta, \pi, d-1)$
    if $gain > bestGain$ **then**
      $bestGain \leftarrow gain$
      $new\mathcal{N} \leftarrow \mathcal{N} \cup \{n\}$
        where $\pi_{a|n} = 1$, $\pi_{last(\mathcal{N})|no'} = 1$
      end if
    end for
end if

is piece-wise linear and convex [17], which implies that the value function of node n is optimal for a polytope of the belief space that includes all reachable beliefs of node n. We can verify that $b_{s|n}$ is a convex combination of the reachable beliefs at node n since it is the normalized version of α_{sn}, which is a discounted sum of reachable beliefs. Hence, whether the beliefs $b_{s|n}$ are reachable or not for each n, Eq. 9, 10 and 11 hold for (globally) optimal controllers.

4 Escaping Local Optima

We describe two algorithms to escape local optima. The first approach does a multi-step forward search to find nodes that can be added to the controller in such a way that EM can resume its progress. The second approach searches for a node to be split in two such that optimizing the parameters of the two new nodes by EM yields the largest improvement.

4.1 Forward Search

Since global optimality is ensured when optimal action and successor node distributions are used for all reachable beliefs, we could perform a forward search from the initial belief to add new nodes each time suboptimal actions or successor

nodes are chosen for some reachable beliefs. However, such a search grows exponentially with the planning horizon. In contrast, EM finds a controller where each action and successor node distribution is optimal according to a forward search of just one step starting from the belief $b_{s|n}$ associated with each n. We propose a technique that gradually extends EM's myopic search by searching increasingly deeper from each $b_{s|n}$. The optimality conditions become stronger with the depth of the search and sufficient in the limit. An infinitely deep search verifies that the controller is optimal for all reachable beliefs while cutting off the search at a finite depth d ensures that the controller is at most $(r_{max} - r_{min})\gamma^d/(1-\gamma)$ away from optimal.

Algorithm 1 describes an incremental forward search technique to escape local optima. The approach verifies whether the action and successor node distributions are optimal with respect to the value function of the controller for all beliefs reachable from some $b_{s|n}$ at increasing depth. When a non-optimal action or successor node choice is detected, a new node is created with optimal action and successor node distributions. We also create nodes for each belief traversed on the path since their action and successor node distributions may change too. These new nodes are added to the controller and the successor node distributions of the existing nodes are perturbed slightly to include an ϵ probability of reaching the new nodes. This is necessary to allow the existing nodes to link to the new nodes since zero probabilities are fixed points in EM.

Note that starting the multi-step search from each $b_{s|n}$ is fine even when $b_{s|n}$ is not a reachable belief as described in the proof of Theorem 2. The benefit of starting the search from each $b_{s|n}$ instead of the initial belief is that a shallower search is often sufficient to find a suboptimal action choice or successor node. Recall that each $b_{s|n}$ is a weighted combination of reachable beliefs that may be arbitrarily deep, hence starting the search from those weighted combinations may significantly reduce the time of the search.

4.2 Node Splitting

Alternatively, we can escape local optima by adapting the HMM state splitting approach to POMDP controllers as described in Algorithm 2. For each node of the controller, consider the possibility of splitting that node in two new nodes n_1 and n_2. To initialize the split, we replace the parameters that involve the node n_{split} being split by parameters that involve n_1 or n_2 in a way that does not change likelihood. More precisely, the parameters $\pi_{a|n_1}$, $\pi_{n'|o'n_1}$, $\pi_{a|n_2}$ and $\pi_{n'|o'n_2}$ are set equal to $\pi_{a|n_{split}}$ and $\pi_{n'|o'n_{split}}$ respectively. As for π_{n_1}, π_{n_2}, $\pi_{n'_1|o'n}$ and $\pi_{n'_2|o'n}$, they are initialized randomly while ensuring that $\pi_{n_1} + \pi_{n_2} = \pi_{n_{split}}$ and $\pi_{n'_1|o'n} + \pi_{n'_2|o'n} = \pi_{n'_{split}|o'n}$. After this neutral initialization of the split, we optimize the parameters by running EM again. To speed up computation, we initialize α and β with those of the unsplit controller. This is similar to keeping the most likely states clamped outside of T_s since α and β are the forward and backward terms that summarize the computation before and after a node split. The M-step adapts the new parameters based on the

Algorithm 2. Node Splitting

Inputs: α, β, π, $iters$
Output: α, β, π (for split controller)
for $n_{split} \in N$ **do**
  split n_{split} into n_1 and n_2
  initialize the splitted controller such that $\pi_{a|n_1} = \pi_{a|n_2} = \pi_{a|n_{split}}$, $\pi_{n'|o'n_1} = \pi_{n'|o'n_2} = \pi_{n'|o'n_{split}}$, $\pi_{n_1} + \pi_{n_2} = \pi_{n_{split}}$ and $\pi_{n'_1|o'n} + \pi_{n'_2|o'n} = \pi_{n'_{split}|o'n}$.
  also split α and β such that $\alpha_{n_1} + \alpha_{n_2} = \alpha_{n_{split}}$, $\beta_{n_1} = \beta_{n_2} = \beta_{n_{split}}$.
  for $i = 1$ to $iters$ **do**
    propagate α and β following Eq. 4 and 5
    perform M-step based on α and β
  end for
  $gain(n)$ = value after M-step
end for
$n^* = \text{argmax}_n\, gain(n)$
assign π, α, β to results of splitting n^*

expectations as usual. After retraining each potential split like this, we select the split which brought the largest increase in likelihood.

4.3 Computational Complexity

We report the computational complexity of EM, node splitting and forward search in Table 1. The complexity of EM is equal to the number of iterations I times the complexity of computing the forward and backward terms in Eq. 4-5, and the expectations in Eq. 1-3. The forward and backward terms are computed recursively by executing t_{max} steps (equal to the planning horizon), where each step can be performed efficiently by variable elimination with a complexity corresponding to the size of the largest factors. Similarly the complexity of the expectations correspond to the largest factors obtained by variable elimination.

The node splitting technique (Alg. 2) evaluates each split by executing EM. Since there are $|\mathcal{N}|$ possible nodes that can be split and the algorithm incrementally grows the controller by splitting one node at a time until there are $|\mathcal{N}|$ nodes, the complexity is $|\mathcal{N}|^2$ times that of EM. As a result, there is a quartic dependency on the size of the controller and this is the dominating factor for large controllers.

The forward search technique (Alg. 1) alternates between EM and adding new nodes based on a recursive search up to some depth d. This search is exponential in d with base $|\mathcal{A}||\mathcal{O}|$, however one can often reduce this complexity by branch-and-bound or sub-sampling. Also, the search is performed by increasing the depth gradually until a non-optimal setting of the parameters is found, at which point the search is terminated. Since we perform a search from each of the $\mathcal{N}$ nodes and the controller is grown incrementally up to $\mathcal{N}$ nodes, the overall complexity of forward search is $|\mathcal{N}|^2$ times that of the recursive search plus $|\mathcal{N}|$ times that of EM. In comparison to node splitting, forward search has a cubic dependency on the size of the controller and therefore tends to scale better for large controllers, but it also has an exponential dependency on the search depth.

Table 1. Computational Complexity. Here I is the number of iterations in EM, t_{max} is the length of the planning horizon, d is the depth of the forward search and $\bar{O}$ is the computational complexity with respect to $|\mathcal{N}|$ and d only.

Forward-Backward step (FB) (Eq. 4 and 5): $O(t_{max}(\lvert\mathcal{N}\rvert\lvert\mathcal{S}\rvert^2 + \lvert\mathcal{N}\rvert^2\lvert\mathcal{S}\rvert)) = \bar{O}(\lvert\mathcal{N}\rvert^2)$ Expectation step (Exp) (Eq. 1-3): $O(\lvert\mathcal{N}\rvert\lvert\mathcal{A}\rvert\lvert\mathcal{S}\rvert^2 + \lvert\mathcal{N}\rvert\lvert\mathcal{A}\rvert\lvert\mathcal{S}\rvert\lvert\mathcal{O}\rvert + \lvert\mathcal{N}\rvert^2\lvert\mathcal{S}\rvert\lvert\mathcal{O}\rvert) = \bar{O}(\lvert\mathcal{N}\rvert^2)$ Expectation-Maximization (EM): $O(I(\text{FB} + \text{Exp})) = \bar{O}(\lvert\mathcal{N}\rvert^2)$
Node splitting (Alg. 2): $O(\lvert\mathcal{N}\rvert^2\text{EM}) = \bar{O}(\lvert\mathcal{N}\rvert^4)$
Recursive search (RS) (Alg. 1): $O(\lvert\mathcal{N}\rvert(\lvert\mathcal{A}\rvert\lvert\mathcal{O}\rvert)^d\lvert\mathcal{S}\rvert^2) = \bar{O}(\lvert\mathcal{N}\rvert(\lvert\mathcal{A}\rvert\lvert\mathcal{O}\rvert)^d)$ Forward search (Alg. 1): $O(\lvert\mathcal{N}\rvert^2\text{RS} + \lvert\mathcal{N}\rvert\text{EM}) = \bar{O}(\lvert\mathcal{N}\rvert^3(\lvert\mathcal{A}\rvert\lvert\mathcal{O}\rvert)^d)$

5 Experiments

We tested four different methods for escaping local optima. The first two are alternatives that add nodes based on a forward search. The third one is the node splitting heuristic. The fourth one is a baseline that uses random restarts for all controller sizes instead of incrementally adding nodes. The detailed settings for these methods are as follows:

- *Forward Search:* Starting from a randomly initialized FSC of $|\mathcal{N}| = |\mathcal{A}|$ nodes, EM is performed till convergence. Then a forward search of increasing depth (up to a time limit) is performed from the belief $b_{s|n}$ associated with each node according to Algorithm 1. As soon as a non-optimal action or successor node distribution is discovered, the search is halted and new nodes for each reachable belief on the current path are added to the controller.
- *Forward Search From Init:* This technique is the same as the previous one except that the forward search is performed only from the initial belief p_s (instead of each $b_{s|n}$. It serves as a baseline to show that forward search from each $b_{s|n}$ is advantageous.
- *Node Splitting:* Starting from a randomly initialized FSC of $|\mathcal{N}| = |\mathcal{A}|$ nodes, we iterate the node splitting procedure of Algorithm 2 until the controller reaches a desired size.
- *Random Restarts:* For each FSC size $|\mathcal{N}| \in \{|\mathcal{A}|, .., n\}$ we randomly initialize a FSC and train it with 500 EM iterations. Note that we also implemented a technique that adds random nodes to escape local optima, but it performed worse than random restarts, so we only report the performance of random restarts as the baseline.

The left column of Figure 1 shows the performance of each method as the number of nodes increases for 6 POMDP benchmarks. Each curve is the median of 21 runs from different initial random controllers with error bars corresponding to

the 25% and 75% quantiles. For the incremental methods, the curves show how the value increases as the number of nodes increases; for the random restarts the results for different $|\mathcal{N}|$ are mutually independent.

The Cheese-Taxi (variant from [12]) and Heaven-and-Hell (variant from [3]) problems are known to have difficult local optima in the sense that the optimal policies include a long sequence of actions such that any small deviation from that sequence is bad. Hence, policy search techniques have trouble finding this sequence by gradually refining a policy since that would involve trying nearby sequences with low values. Only the forward search techniques found good policies because of their ability to modify sequences of actions in one step by adding several nodes at once. Forward search from each $b_{s|n}$ finds good policies with fewer nodes than a forward search from the initial belief because it doesn't have to search as deeply. The random restart and node splitting techniques did not escape the trivial local optima. Since the node splitting technique only changes one node per step, it cannot change multiple actions at once.

The optimal policy of the Chain-of-Chains problem [18] also includes a long sequence of actions, however small deviations are not penalized (i.e., no negative reward), but it will simply take longer for the agent to reach a high rewarding state. As a result, all techniques eventually find the optimal policy, but the forward search techniques find optimal controllers with much fewer nodes, followed by node splitting and random restarts.

The Hallway [10], Hallway2 [10] and Machine [4] problems do not have optimal policies with long sequences of actions that must not be deviated from, but they can still be challenging for policy search techniques that do not explicitly try to escape local optima such as random restarts. For these problems, splitting a node or adding a node based on a short forward search tends to be sufficient to escape local optima. The node splitting technique is generally better because of its ability to evaluate more accurately alternative controllers. Alternative splits are evaluated by re-running EM, which gives a more accurate value than forward search, which adds a node to correct the policy at some reachable belief without re-running EM.

The right column of Figure 1 shows the performance as time increases. Depending on the problem, different techniques among node splitting, forward search and forward search from init may be best. As explained in Section 4.3, forward search tends to scale better than node splitting as the size of the controller increases due to a cubic dependency on the number of nodes (instead of a quartic dependency for node splitting). This is evident for the hallway and chain-of-chains problems. However, forward search may spend a lot of time in a search due to the exponential complexity and it does not necessarily add the best node since it corrects the first non-optimal parameter that it finds. We believe this explains the better performance of node splitting for the machine problem. In general, performing a forward search from each node is advantageous in comparison to a single search from the initial belief since a non-optimal parameter is often found with a shallower search. This explains why forward search performed better for heaven-hell, hallway, hallway2 and machine. However, when a single

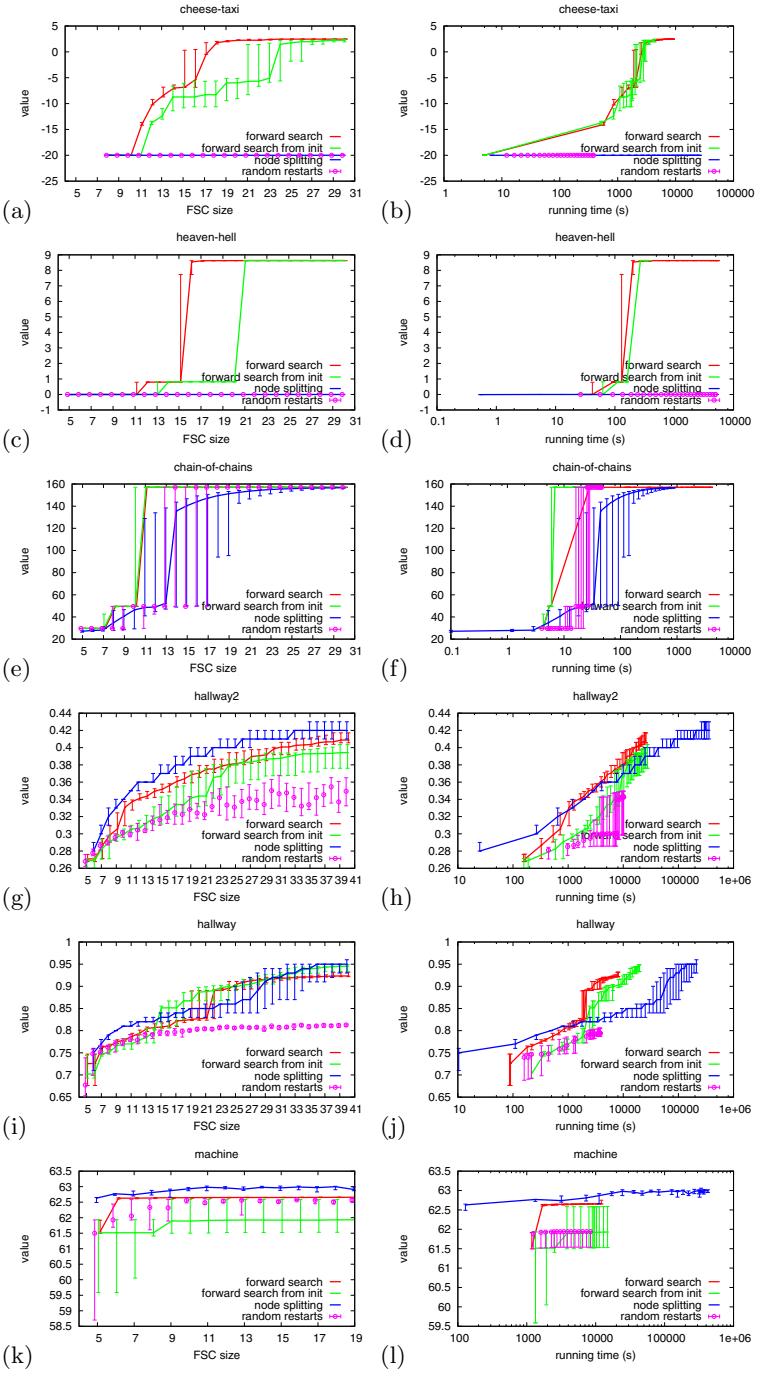


Fig. 1. Performance versus number of nodes (left column) and time (right column). NB: The graphs are best viewed in color.

Table 2. Comparison of the value (average/median over at least 10 runs) for controllers/value functions of different sizes (e.g., # of nodes/α-vectors) indicated in parentheses. Here n.a. indicates that the results are not available and ? indicates that the number of nodes is unknown.

Techniques	cheeseT	heavenH	chainOC	hallway2	hallway	machine
upper bound	2.48	8.64	157.1	0.88	1.18	66.1
SARSOP (10^5 sec)	2.48(168)	8.64(1720)	157.1(10)	0.44(3295)	1.01(4056)	63.2(1262)
SARSOP	-6.38(40)	0.45(55)	**157.1(10)**	0.11(50)	0.15(49)	35.7(42)
biased-BPI+escape	2.13(30)	3.50(30)	40.0(30)	0.41(40)	0.94(40)	63.0(30)
QCLP	n.a.	n.a.	n.a.	n.a.	0.72(8)	61.0(6)
BBSLS	n.a.	7.65(?)	n.a.	n.a.	0.80(10)	n.a.
Forward Search	**2.47(19)**	**8.64(16)**	157.1(11)	0.41(40)	0.92(40)	62.6(19)
Node Splitting	-20.0(30)	0.00(30)	157.1(23)	**0.43(40)**	**0.95(40)**	**63.0(16)**

search from the initial belief does not go deeper than multiple searches from each node, then a single search from the initial belief is faster, which explains the better performance for chain-of-chains.

In Table 2, we compare the forward search and node splitting techniques to a leading point-based value iteration technique (SARSOP [9]) and three policy search techniques for finite state controllers (biased BPI with escape [13], nonlinear optimization (QCLP) [1] and stochastic local search (BBSLS) [3]). The number of nodes was limited to 30 for cheese-taxi, heaven-hell and chain-of-chains, and 40 for hallway2, hallway and machine. Since each node leads to one α-vector in the value function representation, we report the number of alpha-vectors that is closest to 30 and 40 for SARSOP.

Since the optimal policy is not known for several problems, we also report an upper bound on the optimal value (computed by SARSOP) as well as the best value found by SARSOP within 10^5 seconds when the number of α-vectors is not bounded. The results for SARSOP and BPI were obtained by running the APPL package and Poupart's BPI code. The results for QCLP and BBSLS are taken from [1] and [3]. When the number of nodes/α-vectors is limited, forward search achieves the best results for cheese-taxi and heaven-hell, node-splitting achieves the best results for hallway, hallway2 and machine, and SARSOP achieves the best results for chain-of-chains. Overall, forward search obtains the most reliable results by producing controllers that are close to the bests for all problems. SARSOP does not perform well when the number of α-vectors is limited, but can obtain slightly better results when the number of α-vectors is not limited. Note that SARSOP was specifically designed to use fewer α-vectors than other point-based techniques including HSVI2 [16]. Compact controllers/value functions become advantageous for embedded systems with limited memory and processing power. Furthermore, action selection with controllers is instantaneous since there is no computation, whereas a non-trivial search among all α-vectors must be performed to execute policies derived from a set of α-vectors.

6 Conclusion

The main contributions of this paper are a characterization of EM's local optima and the design of two techniques to help EM escape local optima. We showed that EM essentially performs a one-step forward search that optimizes the policy parameters in isolation. Based on this insight, we designed an escape technique that adds new nodes to the controller when a suboptimal action or successor node is detected according to a multi-step forward search that extends EM's implicit one-step search. We also designed a technique to split nodes in two and optimize the new parameters by EM. The forward search technique is the most reliable approach to effectively escape local optima for all 6 benchmark problems, while the node splitting technique finds slightly better controllers for the 3 benchmark problems that do not include a stringent sequence of actions in their optimal policies.

Although there already exist escape techniques for finite state controllers, none of them can be combined with EM (or planning as inference) since they rely on specific information computed by the optimization approaches they were designed for. Hence, assuming that planning as inference will become a leading POMDP solution technique in the near future, this work resolves an important issue by mitigating the effect of local optima and improving the reliability of EM. Our next step is to extend our implementation to factored domains since this is where planning as inference becomes really attractive. In particular, approximate inference techniques such as loopy belief propagation and variational techniques could be integrated within EM and our escape techniques to yield highly scalable algorithms for factored POMDPs.

This work could be extended in several directions, including hierarchical controllers [18], Mealy finite state machines [2] and multi-agent settings [8], which all face local optima, but of different nature than those investigated in this paper.

Acknowledgments. This work was supported by the German Research Foundation (DFG), Emmy Noether fellowship TO 409/1-3 as well as the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

1. Amato, C., Bernstein, D., Zilberstein, S.: Solving POMDPs using quadratically constrained linear programs. In: IJCAI, pp. 2418–2424 (2007)
2. Amato, C., Bonet, B., Zilberstein, S.: Finite-state controllers based on mealy machines for centralized and decentralized POMDPs. In: AAAI (2010)
3. Braziunas, D., Boutilier, C.: Stochastic local search for POMDP controllers. In: AAAI, pp. 690–696 (2004)
4. Cassandra, A.: Exact and approximate algorithms for partially observable Markov decision processes. PhD thesis, Brown University, Dept. of Computer Science (1998)
5. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. J. of the Royal Stat. Society, Series B 39(1), 1–38 (1977)

6. Hansen, E.: An improved policy iteration algorithm for partially observable MDPs. In: NIPS (1998)
7. Hoffman, M., Kueck, H., Doucet, A., de Freitas, N.: New inference strategies for solving Markov decision processes using reversible jump MCMC. In: UAI (2009)
8. Kumar, A., Zilberstein, S.: Anytime planning for decentralized POMDPs using expectation maximization. In: UAI (2010)
9. Kurniawati, H., Hsu, D., Lee, W.S.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proc. Robotics: Science and Systems (2008)
10. Littman, M., Cassandra, T., Kaelbling, L.: Learning policies for partially observable environments: scaling up. In: ICML, pp. 362–370 (1995)
11. Meuleau, N., Kim, K.-E., Kaelbling, L., Cassandra, A.: Solving POMDPs by searching the space of finite policies. In: UAI, pp. 417–426 (1999)
12. Pineau, J.: Tractable Planning Under Uncertainty: Exploiting Structure. PhD thesis, Robotics Institute, Carnegie Mellon University (2004)
13. Poupart, P.: Exploiting structure to efficiently solve large scale partially observable Markov decision processes. PhD thesis, University of Toronto (2005)
14. Poupart, P., Boutilier, C.: Bounded finite state controllers. In: NIPS (2003)
15. Siddiqi, S., Gordon, G., Moore, A.: Fast state discovery for HMM model selection and learning. In: AI-STATS (2007)
16. Smith, T., Simmons, R.: Point-based POMDP algorithms: improved analysis and implementation. In: UAI (2005)
17. Sondik, E.: The optimal control of partially observable decision processes over the infinite horizon: Discounted cost. Operations Research 26(2), 282–304 (1978)
18. Toussaint, M., Charlin, L., Poupart, P.: Hierarchical POMDP controller optimization by likelihood maximization. In: UAI (2008)
19. Toussaint, M., Harmeling, S., Storkey, A.: Probabilistic inference for solving (PO)MDPs. Technical Report EDI-INF-RR-0934, School of Informatics, University of Edinburgh (2006)
20. Toussaint, M., Storkey, A.J.: Probabilistic inference for solving discrete and continuous state Markov decision processes. In: ICML (2006)
21. Vlassis, N., Toussaint, M.: Model free reinforcement learning as mixture learning. In: ICML (2009)

Abductive Plan Recognition by Extending Bayesian Logic Programs

Sindhu Raghavan and Raymond J. Mooney

Department of Computer Science, University of Texas,
1616 Guadalupe, Suite 2.408, Austin TX 78701, USA
{sindhu,mooney}@cs.utexas.edu

Abstract. Plan recognition is the task of predicting an agent's top-level plans based on its observed actions. It is an abductive reasoning task that involves inferring cause from effect. Most existing approaches to plan recognition use either first-order logic or probabilistic graphical models. While the former cannot handle uncertainty, the latter cannot handle structured representations. In order to overcome these limitations, we develop an approach to plan recognition using Bayesian Logic Programs (BLPs), which combine first-order logic and Bayesian networks. Since BLPs employ logical deduction to construct the networks, they cannot be used effectively for plan recognition. Therefore, we extend BLPs to use logical abduction to construct Bayesian networks and call the resulting model Bayesian Abductive Logic Programs (BALPs). We learn the parameters in BALPs using the Expectation Maximization algorithm adapted for BLPs. Finally, we present an experimental evaluation of BALPs on three benchmark data sets and compare its performance with the state-of-the-art for plan recognition.

1 Introduction

Plan recognition is the task of predicting an agent's top-level plans based on its observed actions. It is an abductive reasoning task that involves inferring cause from effect [8]. Traditionally, plan-recognition approaches have been based on first-order logic in which a knowledge-base of plans and actions is developed for the domain and then default reasoning [15] or logical abduction [24] is used to predict the best plan based on the observed actions. However, these approaches are unable to handle uncertainty in the observations or background knowledge and are incapable of estimating the likelihood of different plans. An alternative approach to plan recognition is to use probabilistic methods such as Abstract Hidden Markov Models [5] or statistical n-gram models [2]. While these approaches handle uncertainty, they cannot handle structured representations as they are essentially propositional in nature. As a result, it is also difficult to incorporate planning domain knowledge in these approaches.

The main focus of this paper is to develop an approach to plan recognition that overcomes the limitations described above. Recently, a number of formalisms that integrate both first-order logic and probabilistic graphical models have been developed in the area of *statistical relational learning* (SRL) [11]. Since these formalisms combine the strengths of both approaches, they are well suited for solving problems like plan recognition. We explore the application of one such formalism to plan recognition.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 629–644, 2011.

Of the various SRL formalisms that have been developed, Markov Logic Networks (MLNs), [29], which combine first-order logic and undirected graphical models (Markov nets) have been used for abductive plan recognition by Kate and Mooney [14]. Since MLNs employ deduction for logical inference, they adapt MLNs for abduction by adding reverse implications for every rule in the knowledge base. However, the addition of these rules increases the size and complexity of the MLN, resulting in a computationally expensive model.

In this paper, we explore the application of Bayesian Logic Programs (BLPs) [16], which combine first-order Horn logic and Bayesian networks to plan recognition. BLPs use SLD resolution to generate proof trees, which are then used to construct a ground Bayes net for a given query. However, deduction is unable to construct proofs for abductive problems such as plan recognition. Therefore, we extend BLPs to use logical abduction to construct proofs. In logical abduction, missing facts are assumed when necessary to complete proof trees, and we use the resulting abductive proof trees to construct Bayes nets. We call the resulting model Bayesian Abductive Logic Programs (BALPs). Like all SRL formalisms, BALPs combine the strengths of both first-order logic and probabilistic graphical models, thereby overcoming the limitations of traditional plan recognition approaches mentioned above.

First, we present the necessary enhancements to BLPs to support abduction. Next, we discuss how to learn the parameters in BALPs using the Expectation Maximization algorithm adapted for BLPs [18]. Finally, we present an experimental evaluation of BALPs on three benchmark data sets for plan-recognition and compare its performance with the state-of-the-art.

2 Background

2.1 Logical Abduction

In a logical framework, abduction, is usually defined as follows [28]:

- **Given:** Background knowledge B and observations O, both represented as sets of formulae in first-order logic, where B is typically restricted to a set of Horn clauses and O to a conjunction of ground literals.
- **Find:** A hypothesis H, also a set of logical formulae, such that $B \cup H \not\models \bot$ and $B \cup H \models O$.

Here $\models$ stands for logical entailment and $\bot$ for false, i.e. find a set of assumptions that is consistent with the background theory and explains the observations. There are generally many hypotheses H that explain a particular set of observations O. Following Occam's Razor, the best hypothesis is typically selected based on minimizing $|H|$.

2.2 Bayesian Logic Programs

Bayesian logic programs (BLPs) [16] can be viewed as templates for constructing *directed* graphical models (Bayes nets). Given a knowledge base as a special kind of logic program, standard logical deduction (SLD resolution) is used to automatically construct a Bayes net for a given problem. More specifically, given a set of facts and a query, all

possible Horn-clause proofs of the query are constructed and used to build a Bayes net for answering the query. Standard probabilistic inference techniques are then used to compute the most probable answer.

More formally, a BLP consists of a set of *Bayesian clauses*, definite clauses of the form $A|A_1, A_2, A_3, ...A_n$, where $n \geq 0$ and A, A_1, A_2, A_3,.....,A_n are *Bayesian predicates* (defined below). A is called the head of the clause ($head(c)$) and (A_1, A_2, A_3,....,A_n) is the body ($body(c)$). When $n = 0$, a Bayesian clause is a fact. Each Bayesian clause c is assumed to be universally quantified and range restricted, i.e $variables\{head\} \subseteq variables\{body\}$, and has an associated *conditional probability distribution*: $cpd(c) = P(head(c)|body(c))$.

A *Bayesian predicate* is a predicate with a finite domain, and each ground atom for a Bayesian predicate represents a random variable. Associated with each Bayesian predicate is a combining rule such as *noisy-or* or *noisy-and* that maps a finite set of *cpd*s into a single *cpd* [26]. Let A be a Bayesian predicate defined by two Bayesian clauses, $A|A_1, A_2, A_3, ...A_n$ and $A|B_1, B_2, B_3, ...B_n$, where cpd_1 and cpd_2 are their cpd's. Let θ be a substitution that satisfies both clauses. Then, in the constructed Bayes net, directed edges are added from the nodes for each $A_i\theta$ and $B_i\theta$ to the node for $A\theta$. The combining rule for A is used to construct a single cpd for $A\theta$ from cpd_1 and cpd_2. The probability of a joint assignment of truth values to the final set of ground propositions is then defined in the standard way for a Bayes net: P(X) = $\prod_i P(X_i|Pa(X_i))$, where $X = X_1, X_2, ..., X_n$ represents the set of random variables in the network and $Pa(X_i)$ represents the parents of X_i. The *cpd*s for Bayesian clauses can be learned using the methods described by Kersting and De Raedt [18]. Once a ground network is constructed, standard probabilistic inference methods can be used to answer various types of queries [20].

3 Bayesian Abductive Logic Programs

Bayesian Abductive Logic Programs (BALPs) are an extension of BLPs. In plan recognition, the known facts are insufficient to support the derivation of deductive proof trees for the requisite queries. By using *ab*duction, missing literals can be assumed in order to complete the proof trees needed to determine the structure of the ground network. We first describe the abductive inference procedure used in BALPs. Next we describe how probabilistic parameters are specified and how probabilistic inference is performed. Finally, we discuss how parameters can be automatically learned from data.

3.1 Logical Abduction

Let $O_1, O_2, ..., O_n$ be the set of observations. We derive a set of most-specific abductive proof trees for these observations using the method originally proposed by Stickel [32]. The abductive proofs for each observation literal are computed by backchaining on each O_i until every literal in the proof is proven or assumed. A literal is said to be proven if it unifies with some fact or the head of some rule in the knowledge base, otherwise it is said to be assumed. Since multiple plans/actions could generate the same observation, an observation literal could unify with the head of multiple rules in the knowledge base. For such a literal, we compute alternative abductive proofs. The resulting abductive

Algorithm 1. AbductionBALP

Inputs: Background knowledge KB and observations O_1, O_2, O_3, ..., O_n both represented as sets of formulae in first-order logic, where KB is typically restricted to a set of Horn clauses and each O_i is a ground literal.

Output: Abductive proofs for all O_i.

```
Let Q be a queue of unproven atoms, initialized with O_i
while Q not empty do
    A_i ← Remove atom from Q
    for each rule R_i in KB do
        consequent ← Head literal of R_i
        if A_i unifies with consequent then
            S_i ← unify A_i and consequent and return substitution
            Replace variables in the body of R_i with bindings in S_i. Each literal in the
            body of R_i is a new subgoal.
            for each literal_i in body of R_i do
                if literal_i unifies with head of some rule R_j in KB then
                    add literal_i to Q
                else if literal_i unifies with an existing fact then
                    Unify and consider the literal to be proved
                else
                    if literal_i unifies with an existing assumption then
                        Unify and use the assumption
                    else
                        Assume literal_i by replacing any unbound variables that are exis-
                        tentially quantified in literal_i with new Skolem constants.
                    end if
                end if
            end for
        end if
    end for
end while
```

proof trees are then used to build the structure of the Bayes net using the standard approach for BLPs.

The basic algorithm to construct abductive proofs is given in Algorithm 1. The algorithm takes as input a knowledge base (KB) in the form of Horn clauses and a set of observations as ground facts. It outputs a set of abductive proof trees by performing logical abduction on the observations. These proof trees are then used to construct the Bayesian network. For each observation O_i, AbductionBALP searches for rules whose consequents unify with O_i. For each such rule, it computes the substitution from the unification process and substitutes variables in the body of the rule with bindings from the substitution. The literals in the body now become new subgoals in the inference process. If these new subgoals cannot be proved, i.e if they cannot unify with existing facts or with the consequent of any rule in the KB, then they are assumed. In order to minimize the number of assumptions, the assumed literals are first matched with

(a) Partial Knowledge Base:
Shopping
1. inst(?g,going) | inst(?b,shopping), go-step(?b,?g).
2. inst(?sp,shopping-place) | inst(?s,shopping), store(?s,?sp).
Robbing
3. inst(?p,going) | inst(?r,robbing), go-step(?r,?p).

(b) Observations:
inst(go1,going)
inst(store1,shopping-place)

(c) Ground Abductive Clauses:
inst(go1,going) | inst(a1,shopping), go-step(a1,go1).
inst(go1,going) | inst(a1,robbing), go-step(a1,go1).
inst(store1,shopping-place) | inst(a1,shopping), store(a1,store1).

Fig. 1. (a) A partial knowledge base from the Story Understanding data set. All variables start with "?". (b) The logical representation of the observations. (c) The set of ground rules obtained from logical abduction.

existing assumptions. If no such assumption exists, then any unbound variables in the literal that are existentially quantified are replaced by Skolem constants.

In SLD resolution, which is used in BLPs, if any subgoal literal cannot be proven, the proof fails. However, in BALPs, we assume such literals and allow proofs to proceed till completion. Note that there could be multiple existing assumptions that could unify with subgoals in Step 15. However, if we used all ground assumptions that could unify with a literal, then the size of the ground network would grow exponentially, making probabilistic inference intractable. In order to limit the size of the ground network, we unify subgoals with assumptions in a greedy manner, i.e when multiple assumptions match with a subgoal, we just randomly pick one of them and do not pursue the others. We found that this approach worked well for plan-recognition. For other tasks, domain-specific heuristics could potentially be used to reduce the size of the network.

We now illustrate the abductive inference process with a simple example from the Story-Understanding benchmark data set described in Section 4.1. Consider the partial knowledge base and set of observations given in Figure 1a and Figure 1b respectively. There are two top-level plans, shopping and robbing, in the knowledge base. Note that the action literal "inst(?g, going)" could be observed as part of both shopping and robbing. For each observation literal in Figure 1b, we recursively backchain to generate abductive proof trees. When we backchain on the literal *inst(go1,going)* using Rule 1, we obtain the subgoals *inst(?b,shopping)* and *go-step(?b,go1)*. These subgoals become assumptions since no observations or heads of clauses unify with them. Since $?b$ is an existentially quantified variable, we replace it with a Skolem constant $a1$ to obtain the ground assumptions *inst(a1,shopping)* and *go-step(a1,go1)*. We then backchain on literal *inst(go1,going)* using Rule 3 to get subgoals *inst(?r,robbing)* and *go-step(?r,go1)*. We cannot unify *inst(?r, robbing)* with any observation or existing assumptions; however, we can unify *go-step(?r,go1)* with an existing assumption *go-step(a1,go1)*, thereby binding ?r to a1. In order to minimize the number of assumptions, we first try to match

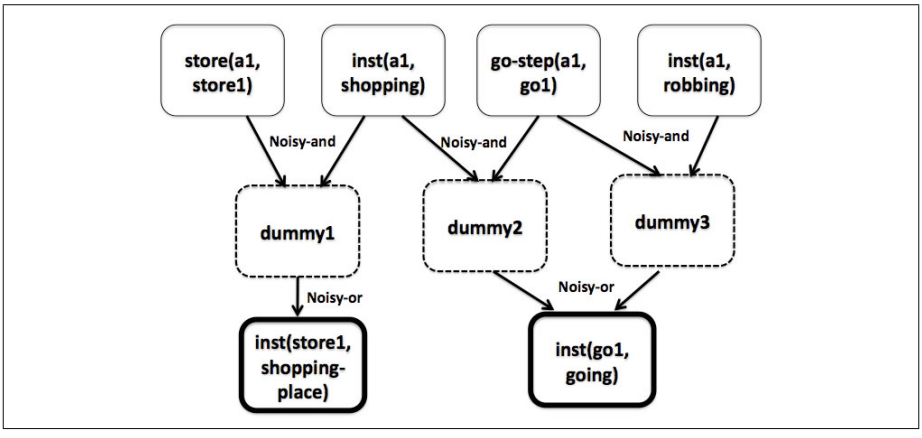


Fig. 2. Bayesian network constructed for example in Figure 1. The nodes with thick borders represent observed actions, the nodes with dotted borders represent intermediate nodes used to combine the conjuncts in the body of a clause, and the nodes with thin borders represent plan literals.

literals with unbound variables to existing assumptions, rather than instantiating them with new Skolem constants. Finally, we backchain on the literal *inst(store1,shopping-place)* using Rule 2 to get subgoals *inst(?s,shopping)*, *store(?s,store1)*. Here again, we match *inst(?s, shopping)* to an existing assumption *inst(a1,shopping)*, thereby binding ?s to a1.

Figure 1c gives the final set of ground rules generated by abductive inference. After generating all abductive proofs for all observation literals, we construct a Bayesian network. Figure 2 shows the Bayesian network constructed for the example in Figure 1. Note that since there are no observations/facts that unify with the subgoals (*inst(?b,shopping)*, *go-step(?b,?g)*, *inst(?r,robbing)*, *go-step(?r,?p)*, and *store(?s,?sp)*) generated during backchaining on observations, SLD resolution will fail to generate proofs. This is typical in plan recognition, and as a result, we cannot use BLPs for such tasks.

The only difference between BALPs and BLPs lies in the logical inference procedure used to construct proofs. Once the abductive proofs are generated, BALPs use the same procedure as BLPs to construct the Bayesian network. We further show in Section 3.3 and Section 4.3 that techniques developed for BLPs for learning parameters can also be used for BALPs.

3.2 Probabilistic Parameters and Inference

We now discuss how parameters are specified in BALPs. We use noisy/logical-and and noisy-or models to specify the *cpd*s in the ground Bayesian network as these models

compactly encode the *cpd* with fewer parameters, i.e. just one parameter for each parent node. Depending on the domain, we use either a strict *logical-and* or a softer *noisy-and* model to specify the *cpd* for combining evidence from the conjuncts in the body of a clause. We use a noisy-or model to specify the *cpd* for combining the disjunctive contributions from different ground clauses with the same head. Figure 2 shows the noisy-and and noisy-or nodes in the Bayesian network constructed for the example in Figure 1.

Given the constructed Bayesian network and a set of observations, we determine the best explanation using standard methods for computing the Most Probable Explanation (MPE) [26], which determines the joint assignment of values to the unobserved nodes in the network which has the maximum posterior probability given the observations. To compute multiple alternative explanations, we use the k-MPE algorithm [25] as implemented in Elvira [10]. For other types of exact probabilistic inference (marginal and joint) we use Netica,[1] a commercial Bayes-net software package.

When the complexity of the ground network makes exact inference intractable (as in the Monroe dataset described in Sect. 4), we have to resort to approximate inference. Due to the (noisy/logical) *and* and *or* nodes in the network, there are a number of deterministic constraints, i.e. 0 values in the *cpd*s. As a result, generic importance sampling algorithms like likelihood weighting used in Elvira failed to generate sufficient samples. Hence, we used SampleSearch [12], an approximate sampling algorithm specifically designed for graphical models with multiple deterministic constraints.

3.3 Parameter Learning

Learning can be used to automatically set the noisy-or and noisy-and parameters in the model. We learn these parameters using the EM algorithm adapted for BLPs by Kersting and De Raedt [18]. In supervised training data for plan recognition, one typically has evidence for the observed actions and the top-level plans. However, we usually do not have evidence for network nodes corresponding to subgoals, noisy-ors, and noisy/logical-ands. As a result, there are a number of variables in the ground networks which are always hidden, and hence EM is appropriate for learning the requisite parameters from the partially observed training data. We simplify the problem by learning only the noisy-or parameters and using a deterministic logical-and model to combine evidence from the conjuncts in the body of a clause. We use uniform priors for top-level plans unless otherwise mentioned.

4 Experimental Evaluation

Unfortunately, there are very few benchmark datasets or rigorous experimental evaluations of plan recognition. In this section, we evaluate BALPs on three plan-recognition datasets that are available. First, we describe experiments to determine if BALPs are more effective for plan recognition than previous approaches. Then, we describe additional experiments evaluating the automatic learning of BALP parameters.

[1] http://www.norsys.com/

4.1 Datasets

Monroe / Reformulated Monroe. The Monroe dataset is an artificially-generated plan-recognition dataset in the emergency response domain by Blaylock and Allen [1]. This domain includes top level plans such as setting up a temporary shelter, clearing a road wreck, and providing medical attention to victims. The task is to infer a single top level plan from a set of observed actions automatically generated by a planner. The planner used is SHOP2 [22] and the domain knowledge is represented as a hierarchical transition network (HTN). We constructed a logical knowledge base representing the domain knowledge encoded in the HTN. We used 1,000 artificially generated examples in our experiments. Each example instantiates one of the 10 top-level plans and contains an average of 10.19 literals describing a sample execution of this plan.

Due to computational complexity, we were unable to compare the performance of BALPs with Kate and Mooney's [14] MLN approach on this domain. Their approach resulted in an MLN with rules containing multiple existentially quantified variables which produced an exponential number of possible groundings, eventually leading to memory overflow. In order to compare BALPs with this MLN approach, we slightly modified the Monroe domain to eliminate this problem without significantly changing the underlying task. The resulting dataset also had 1,000 examples, with an average of 9.7 observations per example. We refer to this dataset as "Reformulated-Monroe."

Linux. The Linux dataset is another plan-recognition dataset created by Blaylock and Allen [3]. Human users were asked to perform various tasks in Linux and their commands were recorded. The task is to predict the correct top level plan from the sequence of executed commands. For example, one of the tasks involves finding all files with a given extension. The dataset consists of 19 top level plans and 457 examples, with an average of 6.1 command literals per example. We constructed the background knowledge base for the Linux dataset based on our knowledge of the commands.

Story Understanding. We also used a dataset[2] that was previously used to evaluate abductive story understanding systems [24,6]. In this task, characters' higher-level plans must be inferred from their actions described in a narrative text. A logical representation of the literal meaning of the text is given for each example. A sample story is: "Bill went to the liquor-store. He pointed a gun at the owner." The plans in this dataset include shopping, robbing, restaurant dining, traveling in a vehicle (bus, taxi or plane), partying and jogging. Most narratives involve more than a single plan. This small dataset consists of 25 development examples and 25 test examples each containing an average of 12.6 literals. We used the background knowledge that was initially constructed for the ACCEL system [24]. Figure 1a and Figure 1b give a partial knowledge base and a partial set of observations from this data set.

Each of these data sets evaluates a distinct aspect of plan recognition systems. Since the Monroe domain is quite large with numerous subgoals and entities, it tests the ability of a plan-recognition system to scale to large domains. On the other hand, the Linux data set is not that large, but since the data comes from real human users, it is quite noisy. There are several sources of noise including cases in which users claim they have

[2] http://www.cs.utexas.edu/~ml/accel.html

successfully executed a top-level plan when actually they have not [2]. Therefore, this data set tests the robustness of a plan-recognition system to noisy input. Monroe and Linux involve predicting a *single* top-level plan; however, in the Story Understanding domain, most examples have multiple top-level plans. Therefore, this data set tests the ability of a plan-recognition system to identify multiple top-level plans.

4.2 Comparison with Other Approaches

We now present comparisons to three previous approaches to plan-recognition across the different benchmark datasets.

Monroe and Linux. We first compared BALPs with Blaylock and Allen's [2] plan-recognition system on both the Monroe and Linux datasets. Their approach learns statistical n-gram models to separately predict plan schemas (i.e. predicates) and their arguments.

We learned the noisy-or parameters for BALPs using the EM algorithm described in Sect. 3.3. We initially set all noisy-or parameters to 0.9, which gave reasonable performance in both domains. We picked a default value of 0.9 for noisy-or parameters based on the intuition that if a parent node is true, then the child node is true with a probability 0.9. We then ran EM with two starting points – random weights and manual weights (0.9). We found that EM initialized with manual weights generally performed the best for both domains, and hence we used this approach for our comparisons. Even though EM is sensitive to starting point, it outperformed other approaches even when initialized with random weights (see Sect. 4.3). Initial experiments found no advantage to using noisy-and instead of logical-and in these domains, so we did not experiment with learning noisy-and parameters.

For Linux, we performed 10-fold cross validation for evaluation and we ran EM until convergence on the training set for each fold. For Monroe, where more data is available, we used 300 examples for training, 200 examples for validation, and the remaining 500 examples for testing. Note that for Monroe, Blaylock and Allen used 4,500 examples for learning parameters. Using 4500 examples for learning BALP parameters results in large training times, and hence we limited to using 300 examples. We ran EM iterations on the training set until the accuracy on the validation set stopped improving. We then used the final learned set of weights to perform plan-recognition on the test set.

For both Monroe and Linux, the plan-recognition task involves inferring a *single* top level plan that best explains the observations. Hence, we computed the marginal probabilities for all ground instantiations of the plan predicates in the network and picked the single plan instantiation with the highest marginal probability.

Due to differences in Blaylock and Allen's experimental methodology and ours, we are only able to directly compare performance using their *convergence score* [2], the fraction of examples for which the correct plan predicate is inferred (ignoring the arguments) when given *all* of the observations.

Table 1 shows the results. BALPs outperform Blaylock and Allen's system on the convergence score in both domains and the difference in the performances was

Table 1. Convergence scores for BALPs and Blaylock and Allen's system for Monroe and Linux. '*' indicates that the difference is statistically significant.

	BALPs	Blaylock and Allen
Monroe	**98.4**	94.2*
Linux	**46.6**	36.1*

statistically significant ($p < .05$) as determined by unpaired t-test [3]. We treated the convergence score as the mean of a Bernoulli variable (schema prediction event) and computed variance accordingly, and then used the mean, variance, and sample size to perform an unpaired t-test. The convergence score for Blaylock and Allen's system on Monroe is already quite high, leaving little room for improvement. However, BALPs was still able to improve over this score by 4.5%. On the other hand, the baseline convergence score for Linux was fairly low, and BALPs were able to improve the results by a remarkable 29.1%. Despite this improvement, the overall convergence score for Linux is not that high. Noise in the data is one reason for the modest score. Another issue with this data set is the presence of very similar plans, like find-file-by-ext and find-file-by-name. The commands executed by users in these two plans are nearly identical, making it difficult for a plan recognition system to distinguish them [3].

Reformulated-Monroe. We also compared the performance of BALPs with Kate and Mooney's [14] MLN approach on the Reformulated-Monroe dataset. For MLNs, we were unable to effectively learn clause weights on this dataset since it was intractable to run Alchemy's existing weight-learners due to the sizes of the MLN and data. Hence, we manually set the weights using the heuristics described by Kate and Mooney [14]. To ensure a fair comparison, we also used manual instead of learned weights for BALPs. We uniformly set all noisy-or parameters to 0.9 and used logical-and to combine evidence from conjuncts in the body of a clause, since this gave good performance on the original Monroe data.

We used two different metrics to compare the performance of the two approaches – convergence score and accuracy. We compared the inferred plan with the correct plan to compute the accuracy score. When computing accuracy, partial credit was given for predicting the correct plan predicate with only a subset of its correct arguments. A point was rewarded for inferring the correct plan predicate, then, given the correct predicate, an additional point was rewarded for each correct argument. For example, if the correct plan was $plan_1(a_1, a_2)$ and the inferred plan was $plan_1(a_1, a_3)$, the accuracy was 66.67%.

The observation set for this domain includes all actions executed to implement the top level plan. In order to evaluate performance for partially observed plans, we performed plan recognition given only subsets of the complete action sequence. Specifically, we report results after observing the first 25%, 50%, 75%, and 100% of the executed actions. Table 2 shows the results. "Accuracy-n" is the accuracy when given the first $n\%$ of the observations. BALPs consistently outperform the MLN approach on

[3] We did not have access to scores for individual examples for Blaylock and Allen's system, hence we performed an *un*paired t-test.

Table 2. Comparative results for Reformulated-Monroe, "*" indicates that the difference is statistically significant

	Convergence Score	Accuracy-100	Accuracy-75	Accuracy-50	Accuracy-25
BALP	**99.90**	**97.40**	**66.80**	**32.67**	**9.83**
MLN	79.66*	79.20*	40.51*	19.26*	4.10*

this data set and all differences are statistically significant ($p < .05$) as determined by the Wilcoxon Sign Rank (WSR) test [30]. The convergence score for BALPs demonstrates a large (25.41%) improvement over MLNs. Finally, we would like to note that the computational complexity of the MLN approach prevented us from running it on the Linux dataset.

Story Understanding. On Story Understanding, we compared the performance of BALPs with the MLN approach of Kate and Mooney [14] and ACCEL [24], a logical-abduction system that uses a metric to guide its search for selecting the best explanation. ACCEL can use two different metrics: *simplicity*, which selects the explanation with the fewest assumptions and *coherence*, which selects the explanation that maximally connects the input observations. This second metric is specifically geared towards text interpretation by measuring *explanatory coherence* [23]. Currently, this bias has not been incorporated in either the BALP or MLN approach.

For BALPs, we were unable to learn useful parameters from just 25 development examples. As a result, we set parameters manually in an attempt to maximize performance on the development set. As before, a uniform value of 0.9 for all noisy-or parameters seemed to work well for this domain. Unlike other domains, using logical-and to combine evidence from conjuncts in the body of a clause did not yield high-quality results. Using noisy-and significantly improved the results; so we used noisy-ands with uniform parameters of 0.9. Here again, the intuition was that if parent node was false or turned off, then the child node would also be false or turned off with a probability 0.9. To disambiguate between conflicting plans, we set different priors for high level plans to maximize performance on the development data. For the MLN approach, we used Kate and Mooney's [14] system with their manually-tuned weights for this dataset.

Since multiple plans are possible in this domain, we computed the most probable explanation (MPE) to infer the best set of plans. We compared the inferred plans with the ground truth to compute *precision*, *recall*, and *F-measure* (the harmonic mean of precision and recall). As before, partial credit was given for predicting the correct plan predicate with some incorrect arguments. The observed literals in this data are already incomplete and do not include all of the actions needed to execute a plan, so they were used as is.

Table 3 shows the results. BALPs performed better than both ACCEL-Simplicity and MLNs. With respect to F-measure, BALPs improved over MLNs by 15.57% and over ACCEL-Simplicity by 33.65%. However, ACCEL-Coherence still performed the best. Since the coherence metric incorporates extra criteria specific to story understanding, this bias would need to be included in the probabilistic models to make them more

Table 3. Comparative results for Story Understanding, "*" indicates that the difference wrt BALPs is statistically significant

	BALP	MLN	ACCEL-Simplicity	ACCEL-Coherence
Precision	72.07	67.31	66.45	89.39*
Recall	85.57	68.10*	52.32*	89.39
F-measure	78.24	67.70*	58.54*	89.39*

competitive. However, the coherence metric is specific to narrative interpretation and not applicable to plan recognition in general.

Overall, BALPs outperformed most existing approaches on the existing benchmark data sets, thus demonstrating that BALPs are a very effective approach to plan recognition.

4.3 Parameter Learning Experiments

We now describe additional experiments that were designed to determine if EM can effectively learn BALP parameters in different plan-recognition domains.

Learning Methodology. We used EM as described in Sect. 3.3 to learn noisy-or parameters for the Linux and Monroe domains.[4] We initially set all noisy-or parameters to 0.9. This gives reasonable performance in both domains, so we compare BALPs with learned noisy-or parameters to this default model which we call "Manual-Weights" (MW). For training, we ran EM with two sets of starting parameters – manual weights (0.9) and random values. We call the former "MW-Start" and the latter "Rand-Start". We used the same training and test splits as described in Section 4.2 for both Linux and Monroe. To measure performance, we computed the convergence score and accuracy scores for various levels of observability as described above.

Learning Results. Table 4 shows the results for different models on Linux. MW-Start consistently outperforms MW, demonstrating that parameter learning improves the performance of default BALP parameters on the Linux domain. Rand-Start does marginally better than MW for all but the 50% and 25% levels of partial observability. However, it does not perform as well as MW-Start, showing that learning from scratch is somewhat better than using default parameters but not as effective as starting learning from reasonable default values.

Table 5 shows the results for different models on Monroe. The performance of MW is already so high that there is little room for improvement, at least for the convergence score. As a result, the MW-Start model could not improve substantially over the MW model. The manual parameters seem to be at a (local) optimum, preventing EM from making further improvements on this data. Rand-Start is performing about as well, sometimes a bit better and sometimes a bit worse than MW, demonstrating that starting

[4] We were unable to learn useful parameters for Story Understanding since the mere 25 development examples were insufficient for training.

Table 4. Results for parameter learning on Linux, "*" indicates that the difference wrt the MW model is statistically significant

	Convergence Score	Accuracy-100	Accuracy-75	Accuracy-50	Accuracy-25
MW	39.82	30.41	28.22	21.84	18.34
MW-Start	**46.6***	**36.32***	**34.06***	**25.45***	**19.83***
Rand-Start	41.57	31.4	29.1	20.53	14.55*

Table 5. Results for parameter learning on Monroe, "*" indicates that the difference wrt the MW model is statistically significant

	Convergence Score	Accuracy-100	Accuracy-75	Accuracy-50	Accuracy-25
MW	98.4	79.16	46.06	20.67	7.2
MW-Start	98.4	79.16	44.63*	20.26	7.33
Rand-Start	98.4	79.86*	44.73*	19.7*	10.46*

from random values the system can learn weights that are about as effective as manual weights for this domain. One reason for the high performance of the MW model on Monroe is the lack of ambiguity in the observations, i.e. there are few observed actions that are part of more than one possible plan. Overall, EM was able to automatically learn effective parameters for BALPs.

5 Discussion

We now discuss various aspects of BALPs that may explain their superior performance. As mentioned earlier, Kate and Mooney's MLN approach [14] cannot be applied to large domains like Monroe since the addition of reverse implications results in a computationally expensive model. As opposed to the explosive grounding of rules in MLNs, BALPs use logical abduction in which only those rules that are relevant to the query are included in the ground network. This results in much smaller networks, enabling BALPs to scale well to large domains. Furthermore, the use of logical abduction allows BALPs to use an existing knowledge base that was created for planning without any modification.

When Blaylock and Allen [2] perform instantiated plan recognition, it is done in a pipeline of two separate steps. The first step predicts the plan schema and the second step predicts the arguments given the schema. Unlike their approach, BALPs are able to jointly predict both the plan and its arguments simultaneously. We believe that BALP's ability to perform joint prediction is at least partly responsible for its superior performance. Both BALP and MLN systems use planning knowledge specified in the form of logical clauses, while Blaylock and Allen's system has no access to domain knowledge. We believe that the ability of BALPs to incorporate domain knowledge is also partly responsible for its superior performance.

Blaylock and Allen's system [2] uses 4500 examples to learn reasonable parameters for the Monroe domain. The MLN system by Kate and Mooney is unable to scale

effectively to this domain. On the other hand, BALPs learn effective parameters for this domain from only 300 examples, demonstrating that EM can effectively learn parameters given a reasonable number of examples. Except for the Story Understanding data set, the EM algorithm used in BALPs could learn parameters automatically from data. The inability of EM to learn effective parameters for this data set can be attributed to the lack of a sufficient number of examples. Note that Kate and Mooney's MLN approach was also unable to learn reasonable weights for Story Understanding. Also note that it is possible to learn parameters for Reformulated-Monroe using EM, but we deliberately avoided using learning to ensure a fair comparison with MLNs. Overall, the success of EM in the original Monroe and Linux domains demonstrates that our approach can automatically learn accurate parameters from data.

Overall the results demonstrates that our approach to plan recognition using BALPs is very effective, generally outperforming existing approaches on the three extant benchmark data sets. As mentioned earlier, each data set tests a specific aspect of the system, and BALP's superior performance on these data sets demonstrate that it is a robust approach to plan recognition.

6 Related Work

Charniak and Goldman [7,6] developed an approach to automatically construct Bayesian networks for plan recognition. Their work is similar to BALPs, but special purpose procedures were used to construct the necessary ground networks rather than using a general-purpose probabilistic predicate logic like MLNs, BLPs, or BALPs. Bui [5] has developed an approach for plan recognition based on Abstract Hidden Markov Models, but this approach cannot handle relational data. Several other systems for plan recognition [24,14,2] were already discussed in Section 4.2.

Poole [27] has developed a framework for Horn clause abduction and shows it relations to Bayesian networks, Chen *et. al* [9] extend Stochastic Logic Programs [21] to incorporate abduction, and Sato [31] has also developed a probabilistic logic called PRISM that performs abduction. Kimmig et. al [19] have developed a method to construct probabilistic explanations using ProbLog. However, none of these approaches have been evaluated on the task of plan recognition. Kersting and De Raedt [17] discuss the differences between BLPs and many of these formalisms; and BALPs inherit these same differences.

7 Future Work

The current comparison to MLNs uses the method of Kate and Mooney [14] without automatic learning of weights. In the future, we would like to explore more efficient online-weight learning [13] with MLNs and compare their performance to BALPs. Furthermore, Kate and Mooney's approach to incorporating logical abduction in MLNs can be improved (c.f. [4]), so comparing to enhanced MLN approaches is another area of future work. We would also like to explore approaches based on lifted inference, which allow to perform probabilistic inference without having to construct ground networks in the future. As mentioned in Section 6, there are several probabilistic logics like Poole's

Horn Abduction, PRISM, and abductive SLPs that can perform abductive reasoning. It would be interesting to apply these frameworks to plan recognition and compare their performance with that of BALPs. In this paper, the background knowledge base was hand-coded; however, we would like to explore techniques that learn abductive knowledge bases automatically from training data.

8 Conclusions

This paper has introduced an approach to plan recognition based on Bayesian Logic Programs (BLPs). We extended BLPs for plan recognition by employing logical abduction to construct Bayesian networks as opposed to the deductive approach currently used in BLPs. We also demonstrated that the model's parameters can be effectively learned using EM. Empirical evaluations on three benchmark data sets demonstrated that the approach generally outperforms the state-of-the-art in plan recognition. We believe that its superior performance is due to its combination of logical abduction, joint probabilistic inference, and incorporation of domain knowledge.

Acknowledgements. We would like to thank Nate Blaylock for sharing Linux and Monroe data sets, Vibhav Gogate for helping us modify SampleSearch algorithm for our experiments, and Parag Singla for his valuable inputs on the paper. This research was funded by MURI ARO grant W911NF-08-1-0242 and Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program. Experiments were run on the Mastodon Cluster, provided by NSF grant EIA-0303609.

References

1. Blaylock, N., Allen, J.: Generating artificial corpora for plan recognition. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 179–188. Springer, Heidelberg (2005)
2. Blaylock, N., Allen, J.: Recognizing instantiated goals using statistical methods. In: Kaminka, G. (ed.) Workshop on MOO 2005, pp. 79–86 (2005)
3. Blaylock, N., Allen, J.F.: Statistical goal parameter recognition. In: ICAPS 2004, pp. 297–305 (2004)
4. Blythe, J., Hobbs, J., Domingos, J., Kate, P., Mooney, R., Implementing, R.: weighted abduction in Markov logic. In: IWCS 2011 (January 2011)
5. Bui, H.H.: A general model for online probabilistic plan recognition. In: IJCAI 2003 (2003)
6. Charniak, E., Goldman, R.: A probabilistic model of plan recognition. In: AAAI 1991, pp. 160–165 (1991)
7. Charniak, E., Goldman, R.P.: A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In: IJCAI 1989, Detroit, MI (1989)
8. Charniak, E., McDermott, D.: Introduction to Artificial Intelligence. Addison, Reading (1985)
9. Chen, J., Muggleton, S., Santos, J.: Learning probabilistic logic models from probabilistic examples. Machine Learning 73(1), 55–85 (2008)
10. Elvira-Consortium: Elvira: An environment for probabilistic graphical models. In: Proceedings of the Workshop on Probabilistic Graphical Models, Cuenca, Spain (2002)

11. Getoor, L., Taskar, B. (eds.): Introduction to Statistical Relational Learning. MIT, Cambridge (2007)
12. Gogate, V., Dechter, R.: Samplesearch: A scheme that searches for consistent samples. In: AISTATS 2007 (2007)
13. Huynh, T.N., Mooney, R.J.: Online max-margin weight learning with Markov logic networks. In: SDM 2011 (2011)
14. Kate, R.J., Mooney, R.J.: Probabilistic abduction using Markov logic networks. In: IJCAI 2009 Workshop on Plan, Activity, and Intent Recognition, Pasadena, CA (July 2009)
15. Kautz, H.A., Allen, J.F.: Generalized plan recognition. In: AAAI, Philadelphia, PA, pp. 32–37 (1986)
16. Kersting, K., De Raedt, L.: Towards combining inductive logic programming with bayesian networks. In: Rouveirol, C., Sebag, M. (eds.) ILP 2001. LNCS (LNAI), vol. 2157, pp. 118–131. Springer, Heidelberg (2001)
17. Kersting, K., De Raedt, L.: Bayesian Logic Programming: Theory and Tool. In: Getoor, L., Taskar, B. (eds.) An Introduction to Statistical Relational Learning. MIT, Cambridge (2007)
18. Kersting, K., Raedt, L.D.: Basic principles of learning Bayesian logic programs. In: Probabilistic Inductive Logic Programming, pp. 189–221 (2008)
19. Kimmig, A., De Raedt, L., Toivonen, H.: Probabilistic explanation based learning. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 176–187. Springer, Heidelberg (2007)
20. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press, Cambridge (2009)
21. Muggleton, S.H.: Learning structure and parameters of stochastic logic programs. In: Matwin, S., Sammut, C. (eds.) ILP 2002. LNCS (LNAI), vol. 2583, pp. 198–206. Springer, Heidelberg (2003)
22. Nau, D., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D., Yaman, F.: Shop2: An HTN planning system. Journal of Artificial Intelligence Research 20, 379–404 (2003)
23. Ng, H.T., Mooney, R.J.: The role of coherence in abductive explanation. In: AAAI 1990, Detroit, MI, pp. 337–442 (July 1990)
24. Ng, H.T., Mooney, R.J.: Abductive plan recognition and diagnosis: A comprehensive empirical evaluation. In: KR 1992, Cambridge, MA, pp. 499–508 (October 1992)
25. Nilsson, D.: An efficient algorithm for finding the M most probable configurations in probabilistic expert systems. Statistics and Computing 8, 159–173 (1998)
26. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, MKP, CA (1988)
27. Poole, D.: Probabilistic Horn abduction and Bayesian networks. Artificial Intelligence 64, 81–129 (1993)
28. Pople, H.E.: On the mechanization of abductive logic. In: IJCAI 1973, pp. 147–152 (1973)
29. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62, 107–136 (2006)
30. Rosner, B.: Fundamentals of Biostatistics. Duxbury Press (2005)
31. Sato, T.: A statistical learning method for logic programs with distribution semantics. In: ICLP 1995, pp. 715–729. MIT Press, Cambridge (1995)
32. Stickel, M.E.: A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation. Tech. Rep. Tech. Note 451, SRI International, CA (September 1988)

Higher Order Contractive Auto-Encoder

Salah Rifai[1], Grégoire Mesnil[1,2], Pascal Vincent[1], Xavier Muller[1], Yoshua Bengio[1], Yann Dauphin[1], and Xavier Glorot[1]

[1] Dept. IRO, Université de Montréal. Montréal (QC), H2C 3J7, Canada
[2] LITIS EA 4108, Université de Rouen. 768000 Saint Etienne du Rouvray, France

Abstract. We propose a novel regularizer when training an auto-encoder for unsupervised feature extraction. We explicitly encourage the latent representation to contract the input space by regularizing the norm of the Jacobian (analytically) and the Hessian (stochastically) of the encoder's output with respect to its input, at the training points. While the penalty on the Jacobian's norm ensures robustness to tiny corruption of samples in the input space, constraining the norm of the Hessian extends this robustness when moving further away from the sample. From a manifold learning perspective, balancing this regularization with the auto-encoder's reconstruction objective yields a representation that varies most when moving along the data manifold in input space, and is most insensitive in directions orthogonal to the manifold. The second order regularization, using the Hessian, penalizes curvature, and thus favors smooth manifold. We show that our proposed technique, while remaining computationally efficient, yields representations that are significantly better suited for initializing deep architectures than previously proposed approaches, beating state-of-the-art performance on a number of datasets.

Keywords: Unsupervised feature learning, deep learning, manifold.

1 Introduction

Good techniques for learning a single layer of useful nonlinear feature extractors appear to be a fundamental ingredient behind most recent successes in training deeper architectures [1]. Many algorithms have already been investigated in this role, starting with the Restricted Boltzmann Machines (RBM) used to initialize ("pre-train") individual layers of Deep Belief Networks [10] and Deep Boltzmann Machines [18]. Alternatives that have been used successfully as learned nonlinear feature extractors include kernel PCA [6], semi-supervised embedding [25], sparse coding[1] [13], classical auto-encoders [2] and novel, better-suited variations of auto-encoders, such as sparse auto-encoders [14, 11, 8], and the Denoising Auto-Encoders (DAE) of [22, 23, 20].

[1] Note that in sparse coding, the forward feature mapping is not computed by a "simple" function, but is the result of an optimization procedure. It is nevertheless a deterministic mapping, albeit a computationally intensive one.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 645–660, 2011.

When used as deterministic feature extractors, both the Restricted Boltzmann Machines and the various flavors of auto-encoders, traditionally yield a mapping of the same basic form: extracted features are a linear projection of the input, passed through a sigmoid nonlinearity. Now all these approaches can easily be extended to other forms of nonlinear mappings, and the question of the relative merits of different types of nonlinear mappings is indeed an important one. But another equally important question that motivated the present study is what algorithm and associated *learning principle* will extract the "best" possible mapping of that traditional form. "Best" is to be understood here in the sense of producing a representation better suited to subsequent processing stages, i.e. extracting relevant, useful, features. This is typically measured objectively by the classification performance of a subsequently built classifier, starting from the representation obtained by unsupervised learning. It can also often be analyzed qualitatively by looking at the linear filters learned by the algorithm.

Modern successes of this kind of unsupervised feature learning approaches appear to depart from the past focus on dimensionality reduction. Quite the opposite, they embrace rich over-complete representations of higher dimensionality than the input. In the context of auto-encoders, no longer having a dimensionality bottleneck means one has to use some other form of regularization to preclude trivial useless solutions (where reconstruction error would be small not only for training examples but for any input configuration). Simple traditional weight decay regularization, which embodies a prior preference toward smaller magnitude weights[2] does not appear to offer an appropriate cure [23]. Some successful variants encourage sparse representations [15, 8], or in the case of the DAE [22, 23], stochastically corrupt the auto-encoder's input, thus changing the objective to that of denoising.

Recently a novel approach for regularizing auto-encoders was proposed, termed Contractive Auto-Encoders (CAE), [17, 16], showing significant improvements in state-of-the-art performance on a number of benchmark datasets. It shares a similar motivation to the DAE of [23]: aiming for robustness to small variations of the input. But the CAE achieves this in a rather different manner: instead of stochastically corrupting the input, it balances the reconstruction error with an analytical penalty term that penalizes the Frobenius norm of the encoder's Jacobian at training points. Another important difference is that the CAE aims directly at obtaining a robust *representation*, whereas the DAE's criterion is to have a robust *reconstruction* of the uncorrupted example. [17] further provide empirical evidence that the trade-off between reconstruction error and the CAE's regularization term yields a representation that captures the local directions of variation dictated by the data, which often correspond to a lower-dimensional non-linear manifold, while being more invariant to the vast majority of directions orthogonal to the manifold.

The present work extends the CAE approach by proposing a simple and computationally efficient technique to not only penalize the first order derivative (Jacobian) of the mapping but also the second order (Hessian) thus furthering

[2] Thus encourages staying closer to a linear mapping.

the stability of the learned representation around training points, which we find to improve the representation both in terms of filters learned (more of the visually interesting ones) and in terms of classification error. While the analytical computation of the Jacobian's norm is no more costly than computing the reconstruction error, an exact analytical computation of the Hessian would be prohibitive. We will retain computational efficiency by using a stochastic approximation of the Hessian's norm.

2 Considered Framework

2.1 Setup and Notation

We are interested in learning a mapping function f that maps an input $x \in \mathbb{R}^{d_x}$ to a representation $h = f(x) \in \mathbb{R}^{d_h}$. We will be using the following notation conventions:

- Wx denotes the matrix vector product between a matrix W and vector x (vectors are considered column vectors by default).
- $\langle A, B \rangle$ denotes the inner product defined as the sum over all elements of the element-wise product. This corresponds to the ordinary dot product for vectors and to the Frobenius product for matrices.
- $\|A\| = \sqrt{\langle A, A \rangle}$, corresponds to the Euclidean norm for vectors and the Frobenius norm for matrices or tensors.
- $J_f(x) = \frac{\partial f}{\partial x}(x)$ denotes the $d_h \times d_x$ Jacobian matrix of f evaluated at x.
- $H_f(x) = \frac{\partial J}{\partial x}(x) = \frac{\partial^2 f}{\partial x^2}(x)$ denotes the $d_h \times d_x \times d_x$ Hessian tensor of f evaluated at x.
- $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ indicates ϵ is a random vector variable following an isotropic Gaussian distribution of variance σ^2.
- $\mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2 I)}[g(\epsilon)]$ denotes the expectation of the enclosed expression with respect to the specified variable and distribution. When unambiguous, we may simply write $\mathbb{E}[g(\epsilon)]$.
- $D_n = \{x^{(1)}, \ldots, x^{(n)}\}$ is a training set of n points $x^{(i)} \in \mathbb{R}^{d_x}$ from which we want to learn mapping f.
- Function f is parameterized by a set of parameters θ. These will be learned by approximately optimizing an objective function $\mathcal{J}$, i.e. $\theta^* = \arg\min_\theta \mathcal{J}(\theta; D_n)$. This approximate optimization will be carried out with a stochastic gradient descent technique.

2.2 Basic Auto-Encoder

The basic Auto-Encoder (AE) framework considered here starts from an encoding function f that maps an input $x \in \mathbb{R}^{d_x}$ to hidden representation $h(x) \in \mathbb{R}^{d_h}$. It has the form

$$h = f(x) = s(Wx + b_h), \tag{1}$$

where s is the logistic sigmoid *activation function* $s(z) = \frac{1}{1+e^{-z}}$. The encoder is parametrized by a $d_h \times d_x$ weight matrix W, and a bias vector $b_h \in \mathbb{R}^{d_h}$.

A decoder function g then maps hidden representation h back to a reconstruction y:

$$y = g(h) = s(W'h + b_y), \tag{2}$$

The decoder's parameters are a bias vector $b_y \in \mathbb{R}^{d_x}$, and a matrix W'. In this paper we only explore the tied weights case, in which $W' = W^T$.

Basic auto-encoder training consists in finding parameters $\theta = \{W, b_h, b_y\}$ that minimize the reconstruction error on a training set of examples D_n, i.e. minimizing the following objective function:

$$\mathcal{J}_{\mathrm{AE}}(\theta) = \sum_{x \in D_n} L(x, g(f(x))), \tag{3}$$

where $L(t, r)$ is the reconstruction error between target t and reconstruction r (typically squared error or cross-entropy loss).

2.3 The First-Order Contractive Auto-Encoder

To encourage robustness of $f(x)$ to small variations of a training input x, [17] penalize its *sensitivity* to that input, measured as the Frobenius norm of the Jacobian $J_f(x)$ [16]. Thus a Contractive Auto-Encoder (CAE) is trained to optimize the following objective:

$$\mathcal{J}_{\mathrm{CAE}}(\theta) = \sum_{x \in D_n} L(x, g(f(x))) + \lambda \|J_f(x)\|^2, \tag{4}$$

where λ is a positive hyperparameter that controls the strength of the regularization.

Let $h = f(x)$. The linear+sigmoid mapping yields a simple expression for the penalty term: $\|J_f(x)\|^2 = \sum_{j=1}^{d_h} \|h_j(1 - h_j)W_j\|^2$. This has a similar computational cost as computing the reconstruction error (e.g. squared error is $\| - x + b_y + \sum_{j=1}^{d_h} h_j W_j\|^2$). Thus computing the objective and the gradient update in a CAE is only about twice as expensive as in an ordinary AE; both have the same overall computational complexity of $O(d_h d_x)$.

2.4 Proposed Higher Order Regularization

The penalty over the Jacobian yields a preference for mappings f that are invariant locally at the training points. We propose to extend the flat region further away from the training points by also penalizing higher order terms, in particular the curvature, characterized by the Hessian.

While computing the Jacobian regularization is essentially no more expensive than computing the reconstruction error, computational requirements for penalizing analytically computed higher orders grows exponentially with the order. Specifically, computing the norms of k^{th} order derivative of f has a computational complexity of $O(d_h d_x^k)$. Computing the gradient of such higher order regularization terms with respect to model parameters thus becomes quickly prohibitive.

We propose instead to use a stochastic approximation of the Hessian Frobenius norm. Consider a noise random variable $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, we have

$$\|H_f(x)\|^2 = \lim_{\sigma \to 0} \frac{1}{\sigma^2} \mathbb{E}\left[||J_f(x) - J_f(x+\epsilon)||^2 \right] \tag{5}$$

This is obtained starting with a Taylor series expansion of J_f around x; the proof is given in the appendix. For non-infinitesimal noise, the right hand side would also contain contributions from higher order derivatives, but these vanish in the limit $\sigma \to 0$.

Our novel proposed algorithm, that we shall call Contractive Auto-Encoder with Hessian regularization (CAE+H) thus tries to optimize the following objective:

$$\mathcal{J}_{\text{CAE+H}}(\theta) = \sum_{x \in D_n} L(x, g(f(x))) + \lambda \, ||J_f(x)||^2 + \gamma \mathbb{E}\left[||J_f(x) - J_f(x+\epsilon)||^2 \right], \tag{6}$$

where λ and γ are non-negative hyperparameters that control how strongly we penalize the Jacobian and the Hessian. Informally, we see that the last term limits the Hessian norm by encouraging the Jacobian norm at x and at neighboring points to be close to zero.

In practice, we will use a stochastic approximation of the expectation by generating a small mini-batch of a few corrupted samples $\tilde{x} = x + \epsilon$ (all from the same x, but different ϵ) thus incurring some variance for the computational benefit of not having to explicitly compute the analytical Hessian.

3 Geometric Interpretation

According to the manifold assumption [5], structured data in a high dimensional space, such as natural images, are thought to concentrate near a lower dimensional non-linear manifold. With this perspective in mind, [17] give the following geometric interpretation of the workings of the CAE. The penalty on the Jacobian norm encourages, at training points, an equally strong contractive mapping in all input space directions. This pressure is however exactly counteracted (at a local minimum) by the gradient of the reconstruction error term, that ensures that distinct training points can be accurately reconstructed from their representation[3]. In particular, neighboring training points on the manifold must receive a distinguishable mapping. This means that in the area surrounding the examples, the learnt mapping has to be far less contractive in directions parallel to the manifold, while it can be maximally contractive in the directions orthogonal to the manifold. This view means that the representation will change little when moving orthogonal to the manifold and most when moving along the manifold,

[3] Note that having tied weights means that encoder and decoder weights have the same magnitude. Consequently the CAE cannot merely play the game of having the encoder scale down the input and the decoder blow it up again.

so that the learnt representation constitutes a kind of coordinate system on the manifold.

[17] provide empirical evidence that this is indeed happening by examining, for the learnt mapping, the singular value spectrum of the Jacobian at training points. It characteristically showed but a few singular values much larger than the others, which confirms that the mapping is maximally contractive in the overwhelming majority of directions (presumed to be those orthogonal to the low dimensional manifold), while being significantly less contractive in only a few directions (those parallel to the low dimensional manifold). From this geometric interpretation, the leading singular vectors of the Jacobian matrix (those associated with large singular values) are the directions in input space to which the representation is most sensitive, and can be understood as *spanning the tangent space of the manifold*.

The addition of the Hessian penalty, introduced here, will encourage the Jacobian to change slowly (or not at all) as we move away from a training point. When the move is orthogonal to the the manifold, this should ensure that the directions to which the representation is most sensitive remain those parallel to the manifold. When the move is along the manifold, forcing the Jacobians at two nearby points to be close means, from the above geometrical interpretation, that the tangent spaces at these points must be close. It thus *prefers flatter manifolds by penalizing curvature*.

4 Related Previous Work

Traditional regularization [21] for learning a mapping f imposes a prior preference over the considered space of functions (or their parameters), through the addition of a penalty term $\lambda\Omega(f)$ to the regression objective. Thus the usual weight decay regularization penalizes the squared norm of the weights: $\Omega_{wd}(f) = \|W\|^2$. In the case of a linear mapping this corresponds precisely to the norm of its Jacobian matrix. But this is no longer the same with a nonlinear mapping. Penalization of second order derivatives (roughness penalty) is also a commonly employed in fitting statistical models. It is used extensively in non-parametric methods such as smoothing splines [24, 9]. These regularizers can all be expressed in the following general form, which penalizes the norm of the k^{th} order derivatives:

$$\Omega_k(f) = \int \left|\left| \frac{\partial^k f(x)}{\partial x^{(k)}} \right|\right|^2 dx,$$

where the integral is over the whole domain of f. This yields a simple tractable expression when applied to a linear mapping or to non-parametric spline models. But when learning a parameterized non-linear mapping such as linear+sigmoid, the integral becomes intractable.

Thus [4] investigated penalizing, *on training data points only*, the norm of the Hessian *diagonal only*, which he computes analytically. This is similar to our approach, except we chose to penalize a stochastic approximation of the whole Hessian norm, rather than an analytically exact diagonal only. Our stochastic

approximation scheme is a simple, practical, and computationally efficient alternative to penalizing the full Hessian norm. Also in [4] the goal was to regularize a supervised objective, which is quite different from our focus on unsupervised feature learning.

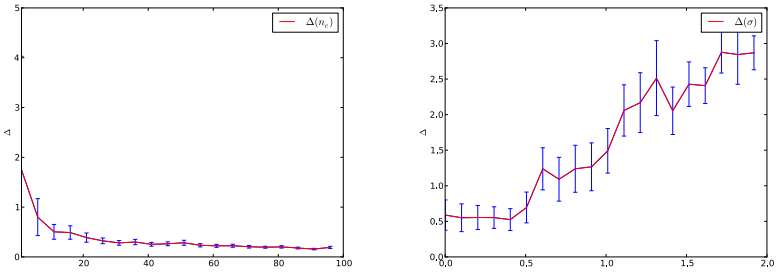
Note that applying a regularizer in this way is a departure from the classical view as a prior on function space. Since the regularization term is evaluated only on or around training points, the regularizer becomes data-dependent, and can no longer be considered a true "prior" in the Bayesian sense. Note that this is true also of the various attempts to yield sparse representations with auto-encoder variants [8, 14, 11] that were inspired by the influential work on sparse coding by [13]. Since the sparsity of representation is sought only at training points, it also constitutes a data-dependent regularizer. Whereas in previous work this restriction to the training points (instead of the whole input domain) might have been seen as an approximation, here this is a *very important feature*. **We only want the contraction and flatness effects where the data density is large.**

The geometrical interpretation of the Jacobian as representing the tangent space of a manifold is also related to the work on tangent propagation [19], semi-supervised embedding [25] and non-local manifold Parzen windows [3], but with marked differences. Tangent propagation [19] uses prior knowledge, based on known transformations, to define the contractive directions, and is used in a supervised setting. The CAE does not use any prior information to choose the directions of contraction. They are implicitly extracted from the dataset during the training. Semi-supervised embedding [25], in conjunction with a supervised criterion, uses pairs of neighboring points and tries to pull them together, thus explicitly contractive in directions *along* the manifold. This is to be contrasted with the CAE that contracts mostly in directions *orthogonal* to the manifold, without the computational requirement of a neighborhood graph. Non-local manifold Parzen windows [3] learns a function to explicitly output tangent directions at a point, whereas the CAE's tangent directions are to be found in the Jacobian matrix of the mapping function it learns. Contrary to the CAE+H that we propose here, none of these methods seem to address the curvature of the modeled manifold.

5 Experiments

5.1 Analysis of CAE+H

Efficiency of the approximation. Following Eq. (5), we approximate stochastically the Frobenius norm of the true Hessian of the hidden layer $h(x)$ with respect to the input. As we have seen earlier, the approximation depends on two hyperparameters, the number of corrupted inputs n_c, and the standard deviation of the Gaussian noise σ. To illustrate how the approximation is affected by different values of these hyperparameters, we estimate the absolute value of the difference Δ between the true Hessian norm and the approximated norm as we vary both hyperparameters. As one can expect, the optimal number of



(a) mini-batch size n_c (b) noise standard deviation σ

Fig. 1. Using the hidden representation $h(x)$ of an auto-encoder, we compute the meanand variance of the norm of the Hessian on 1000 samples with an optimal fixed set of hyperparameters. Evolution of Δ with respect to (a) n_c and (b) σ.

corrupted inputs tends to infinity Figure 1(a), and the optimal variance tends to zero Figure 1(b). It should be noted that while evaluating the exact true Hessian norm is feasible in $O(d_x^2 d_h)$, in practice computing the gradient of the penalty w.r.t model parameters is prohibitive. With our approximation, estimating the norm costs only $O(n_c d_x d_h)$. In our experiments, due to numerical instability we avoid using tiny values for the standard deviation. This also has the beneficial effect of penalizing higher order derivatives.

Penalization of Higher Order Terms. In high dimensional input space, we will use a small number of corrupted samples, as it would be computationally expensive otherwise. Although the approximation is less accurate in this case, Figure 2 shows that using Eq. (5) in the objective cost is still penalizing the norm of the true Hessian in contrast with the simple CAE objective cost on the MNIST dataset. The variance of the noise is also chosen large enough so that higher order terms are also penalized. The values used in our experiments are in range $\sigma \in [0.1, 0.5]$ and $n_c \in \{4, 6, 8\}$.

5.2 Experimental Results

Considered models. In our experiments, we compare the proposed Higher Order Contractive Auto-Encoder (CAE+H) against the following models for unsupervised feature extraction:

- RBM-binary : Restricted Boltzmann Machine trained with Contrastive Divergence,
- AE: Basic auto-encoder,
- DAE: Denoising auto-encoder with Gaussian noise,
- CAE: Contractive auto-encoder.

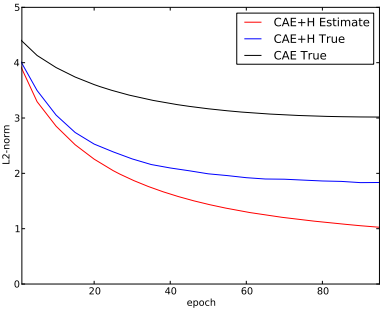


Fig. 2. On MNIST, during the optimization of CAE+H we measure the true Hessian norm and the approximated norm with $n_c = 4$ and $\sigma = 0.3$. We see that our computationnally efficient approximation is very effective at constraining the norm of the true Hessian.

MNIST. We tried our model on the well known digit classification problem (28×28 gray-scale pixel values scaled to [0,1]). We used the usual split of 50000 examples for training, 10000 for validation, and 10000 for test.

To empirically verify the advantage of the representation learnt using the CAE+H with respect to its discriminative power, we pretrained different auto-encoders using the regularization described above and used their hidden representation $h(x)$ as an input for a logistic regression. We also used these representations to initialize a one hidden layer MLP. The auto-encoders were also compared to an RBM.

As discussed in section 3, we illustrate how the CAE+H captures the directions of allowed variations within the data manifold. For any encoding function f, we can measure the *average contraction ratio* for pairs of points, one of which, x_0 is picked from the validation set, and the other x_1 randomly generated on a sphere of radius r centered on x_0 in input space. How this average ratio evolves as a function of r yields a *contraction curve*. We have computed these curves for the models for which we reported classification performance (the contraction curves are however computed with their initial parameters prior to fine tuning). Results are shown in Figure 6 for single-layer mappings.

5.3 MNIST Variants

In addition to MNIST, we used some of its variants, namely MNIST-*rot*(digits with added random rotation) and MNIST-*bg-img*(digits with random image background) consisting of 10000 training, 2000 validation, 50000 test examples [12][4]. Finally, we considered an artificially generated dataset *rect* for shape classification where the task is to discriminate between tall and wide rectangles (white on black).

[4] Datasets available at `http://www.iro.umontreal.ca/~lisa/icml2007`.

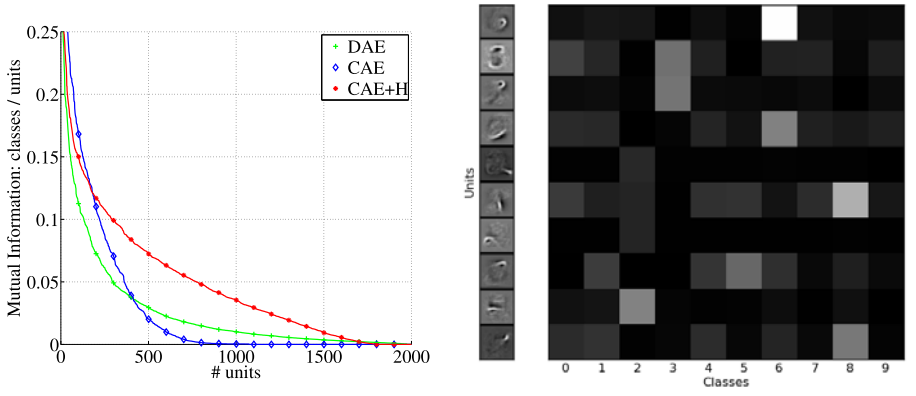


Fig. 3. On MNIST, *Left:* **Mutual information between class labels and individual hidden units** Hidden units were binarized with a threshold of 0.5. CAE+H extracts more discriminant features than others methods. *Right:* **Mutual Information between each class and random chosen hidden units.** E.g The first hidden unit is only responsive for the "6" digits.

Table 1. Comparison of the quality of extracted features from different models when using them as the fixed inputs to a logistic regression (top row) or to initialize a MLP that is fine-tuned (bottom row). Classification error rate is reported together with a 95% confidence interval. CAE+H is clearly less dependent on the fine-tuning step and outperforms all other considered models by the quality of its representation.

Model \ pretrain	AE	RBM	DAE	CAE	CAE+H
LogReg	2.17±0.29	2.04±0.28	2.05±0.28	1.82±0.26	**1.2**±0.21
MLP	1.78±0.26	1.3±0.22	1.18±0.21	1.14±0.21	**1.04**±0.20

5.4 CIFAR-10

In this section, we used the same preprocessing pipeline as [7] for feature extraction. First, we randomly extract 160000 patches from the first 10000 images of CIFAR-10. Each patch is locally contrast-normalized (subtract the mean and divide by its standard deviation) and ZCA whitened.

In order to qualitatively compare filters obtained with different approaches, we trained a CAE+H and other models in an unsupervised fashion with a high number n_h of hidden units on this set of preprocessed patches (see Figure 5). This figure will be discussed in the next section.

Once a non-linear feature-mapping has been learnt by unsupervised training, we can evaluate different feature extraction techniques on a set of labeled images using classification accuracy.

The same convolutional feature extraction as [7] has been used to compare CAE+H and CAE with different models namely Sparse Restricted Boltzmann

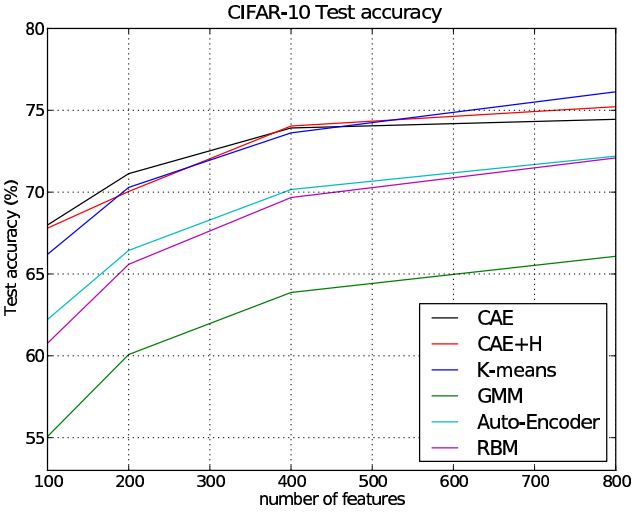


Fig. 4. Test accuracy of CAE and CAE+H versus different learning algorithms [7]

Table 2. Comparison of stacked second order contractive auto-encoders with 1 and 2 layers (CAE+H-1 and CAE+H-2) with other 3-layer stacked models and baseline SVM. Test error rate on all considered classification problems is reported together with a 95% confidence interval. Best performer is in bold, as well as those for which confidence intervals overlap. Clearly CAE+Hs can be successfully used to build top-performing deep networks. 2 layers of CAE+H often outperformed 3 layers of other stacked models.

Data Set	SVM$_{rbf}$	SAE-3	RBM-3	DAE-b-3	CAE-2	CAE+H-1	CAE+H-2
rot	11.11±0.28	10.30±0.27	10.30±0.27	**9.53**±0.26	**9.66**±0.26	10.9±0.27	**9.2**±0.25
bg-img	22.61±0.379	23.00±0.37	16.31±0.32	16.68±0.33	15.50±0.32	15.9±0.32	**14.8**±0.31
rect	2.15±0.13	2.41±0.13	2.60±0.14	1.99±0.12	1.21±0.10	**0.7**±0.07	**0.45**±0.06

Machine, Sparse Auto-encoder, Gaussian Mixtures Model and K-means clustering. On every image, each patch is preprocessed and passed through the encoder to obtain n_h feature maps. Then, to roughly reduce the dimension, features are sum-pooled together over quadrants of the feature maps. So the input dimension of the linear classifier is equal to $4n_h$. We trained a L2-regularized linear SVM on these features and reported the test classification accuracy in Figure 4.

As another experiment, we used the sum pooled-features learned above as the input to a shallow MLP to see if we could improve upon the SVM performance. We used a CAE to initialize the parameters of the MLP. With this method, we were able to achieve a classification accuracy of 78.5% on CIFAR-10.

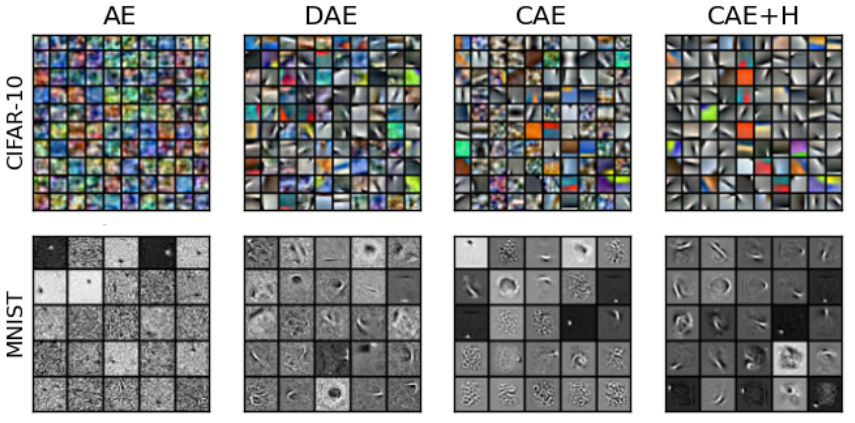


Fig. 5. Random filters from various model types with high dimensional hidden representation learnt on CIFAR10(4000*units*) and MNIST(2000*units*). CAE+H extracts smoother features and obtains a smaller proportion of noisy filters despite the exaggerated overcompleteness of the representation.

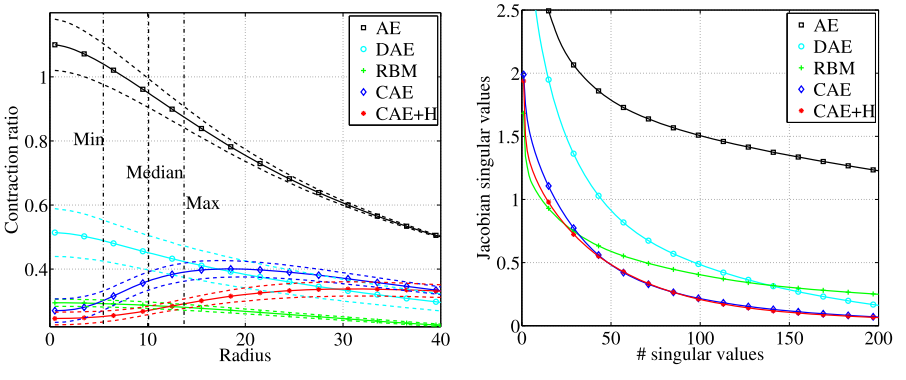


Fig. 6. On MNIST, *Left:* **Contraction ratio with respect to distance from test samples.** The contractive models have a non-monotonic contraction, CAE+H contracts further away from the samples. *Right:* **Averaged Jacobian spectrum across test samples.**

6 Discussion

Upon qualitative examination of Figure 5, we can venture that the simple autoencoder learns poor feature detectors while DAE, CAE and CAE+H appear to

capture more relevant information. CAE-H and DAE present a higher proportion of structured, local and sharp filters than the CAE.

From this observation, we can hypothesize that the sharper-looking filters are likely to be more class-specific. In order to verify this objectively, we measured the mutual information between class labels and individual hidden units binarized with a threshold of 0.5). These results are reported in Figure 3. The CAE+H indeed has more specialized hidden units, that correlate better with specific class labels.

This can explain the much better classification performance obtained with a simple logistic regression stacked on the thus learnt representation, even without any supervised fine tuning of the filters, as we can see in Table 1.

We have proposed an original and computationally efficient way to regularize an auto-encoder by penalizing higher order derivatives of its learnt mapping, without having to explicitly compute them. This novel unsupervised method for learning feature detectors was shown to learn more appropriate features for supervised tasks than several recently proposed competing methods. In particular, it allowed us to beat the state-of-the-art on MNIST and variants, and to reach it on CIFAR-10.

Compared to the other considered approaches, the CAE+H doesn't seem to depend as much on a supervised fine-tuning step to yield good performance. This is especially interesting if one wants to apply the technique to build deep models by stacking. Indeed the fine-tuning step is subject to the vanishing gradient problem during back-propagation. So it is likely to be of great benefit to not depend so much on the fine-tuning step.

Appendix

Proof that $\lim_{\sigma \to 0} \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \sigma^2 I)} \left[\frac{1}{\sigma^2} \|J(x+\epsilon) - J(x)\|^2 \right] = \|H(x)\|^2$

Let $\epsilon \in \mathbb{R}^{d_x}$. Taylor series expansion of the Jacobian around x yields

$$J(x+\epsilon) = J(x) + \left(\sum_{i=1}^{d_x} \epsilon_i \frac{\partial J}{\partial x_i}(x) \right) + R(x, \epsilon), \tag{7}$$

where the remainder $R(x, \epsilon)$ contains all higher order expansion terms:

$$R(x, \epsilon) = \sum_{K=2}^{\infty} \frac{1}{K!} \sum_{i_1, \ldots, i_K} \epsilon_{i_1} \ldots \epsilon_{i_K} \frac{\partial^K J}{\partial x_{i_1} \ldots \partial x_{i_K}}(x).$$

We can thus write for a given $\sigma \in \mathbb{R}$:

$$\frac{1}{\sigma^2}\left\|J(x+\epsilon)-J(x)\right\|^2 = \frac{1}{\sigma^2}\left\|\left(\sum_{i=1}^{d_x}\epsilon_i\frac{\partial J}{\partial x_i}(x)\right)+R(x,\epsilon)\right\|^2$$
$$= \underbrace{\frac{1}{\sigma^2}\left\|\sum_{i=1}^{d_x}\epsilon_i\frac{\partial J}{\partial x_i}(x)\right\|^2}_{T_1} + \underbrace{\frac{2}{\sigma^2}\left\langle\sum_{i=1}^{d_x}\epsilon_i\frac{\partial J}{\partial x_i}(x)\,,\,R(x,\epsilon)\right\rangle}_{T_2}$$
$$+\underbrace{\frac{1}{\sigma^2}\left\langle R(x,\epsilon)\,,\,R(x,\epsilon)\right\rangle}_{T_3} \tag{8}$$

Let $\epsilon \sim \mathcal{N}(0,\sigma^2 I)$ and let us consider the expectation of subexpressions T_1, T_2, T_3 in the limit $\sigma \to 0$. Remainder R is a sum of terms of the form $c\,\epsilon_{i_1}\ldots\epsilon_{i_K}$ where c is constant with respect to ϵ, and $K \geq 2$. Thus inner product T_2 and T_3 will be sums of products of the same form but where $K \geq 3$. Each such product is a polynomial in components of ϵ, with the smallest exponent being either 1 or ≥ 2. Such polynomials that have at least one odd exponent will have an expectation of 0 (odd moments of a 0-mean Gaussian are zero). Those that contain only even exponents will at least have one component with an exponent of 4 or more or two components with an exponent of at least 2 each, so their expectation will be a polynomial in σ whose smallest exponent is no less than 4. In all cases, for $K \geq 3$, we will have $\lim_{\sigma\to 0}\frac{1}{\sigma^2}\mathbb{E}\left[c\epsilon_{i_1}\ldots\epsilon_{i_K}\right]=0$. Since T_2 and T_3 are sums of terms of this form, we can write

$$\lim_{\sigma\to 0}\mathbb{E}\left[T_2\right] = \lim_{\sigma\to 0}\mathbb{E}\left[T_3\right] = 0. \tag{9}$$

The expectation of the first term T_1 yields

$$\mathbb{E}\left[T_1\right] = \mathbb{E}\left[\frac{1}{\sigma^2}\left\langle\sum_{i=1}^{d_x}\epsilon_i\frac{\partial J}{\partial x_i}(x)\,,\,\sum_{i=1}^{d_x}\epsilon_i\frac{\partial J}{\partial x_i}(x)\right\rangle\right]$$
$$= \frac{1}{\sigma^2}\sum_{i=1}^{d_x}\sum_{j=1}^{d_x}\mathbb{E}\left[\epsilon_i\epsilon_j\right]\left\langle\frac{\partial J}{\partial x_i}(x)\,,\,\frac{\partial J}{\partial x_j}(x)\right\rangle.$$

For $i \neq j$, $\mathbb{E}\left[\epsilon_i\epsilon_j\right] = \mathbb{E}\left[\epsilon_i\right]\mathbb{E}\left[\epsilon_j\right] = 0$ and all the corresponding terms in the above sum vanish. For $i = j$ however we have $\mathbb{E}\left[\epsilon_i\epsilon_j\right] = \mathbb{E}\left[\epsilon_i^2\right] = \sigma^2$. Consequently the above sum reduces to

$$\mathbb{E}\left[T_1\right] = \frac{1}{\sigma^2}\sum_{i=1}^{d_x}\sigma^2\left\langle\frac{\partial J}{\partial x_i}(x)\,,\,\frac{\partial J}{\partial x_i}(x)\right\rangle = \left\|H(x)\right\|^2. \tag{10}$$

Putting together Equations 8, 9 and 10, we can conclude:

$$\lim_{\sigma\to 0}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)}\left[\frac{1}{\sigma^2}\left\|J(x+\epsilon)-J(x)\right\|^2\right] = \lim_{\sigma\to 0}\mathbb{E}\left[T_1\right]+\lim_{\sigma\to 0}\mathbb{E}\left[T_2\right]+\lim_{\sigma\to 0}\mathbb{E}\left[T_3\right]$$
$$= \left\|H(x)\right\|^2.$$

References

[1] Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning 2(1), 1–127 (2009); Also published as a book. Now Publishers (2009)

[2] Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: NIPS, vol. 19, pp. 153–160. MIT Press, Cambridge (2007)

[3] Bengio, Y., Larochelle, H., Vincent, P.: Non-local manifold parzen windows. In: NIPS, vol. 18. MIT Press, Cambridge (2006)

[4] Bishop, C.M.: Curvature-driven smoothing: A learning algorithm for feedforward networks. IEEE Transactions on Neural Networks 5(4), 882–884 (1993)

[5] Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)

[6] Cho, Y., Saul, L.: Kernel methods for deep learning. In: NIPS 2009, pp. 342–350, NIPS Foundation (2010)

[7] Coates, A., Lee, H., Ng, A.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011), JMLR W&CP (2011)

[8] Goodfellow, I., Le, Q., Saxe, A., Ng, A.: Measuring invariances in deep networks. In: NIPS 2009, pp. 646–654 (2009)

[9] Hastie, T., Tibshirani, R.: Generalized Additive Models. Chapman and Hall, Boca Raton (1990)

[10] Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)

[11] Kavukcuoglu, K., Ranzato, M., Fergus, R., LeCun, Y.: Learning invariant features through topographic filter maps. In: Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR 2009), IEEE, Los Alamitos (2009)

[12] Larochelle, H., Erhan, D., Courville, A., Bergstra, J., Bengio, Y.: An empirical evaluation of deep architectures on problems with many factors of variation. In: Ghahramani, Z. (ed.) ICML 2007, pp. 473–480. ACM, New York (2007)

[13] Olshausen, B.A., Field, D.J.: Sparse coding with an overcomplete basis set: a strategy employed by V1? Vision Research 37, 3311–3325 (1997)

[14] Ranzato, M., Poultney, C., Chopra, S., LeCun, Y.: Efficient learning of sparse representations with an energy-based model. In: NIPS 2006 (2007)

[15] Ranzato, M., Poultney, C., Chopra, S., LeCun, Y.: Efficient learning of sparse representations with an energy-based model. In: NIPS 2006, pp. 1137–1144. MIT Press, Cambridge (2007)

[16] Rifai, S., Muller, X., Mesnil, G., Bengio, Y., Vincent, P.: Learning invariant features through local space contraction. Technical Report 1360, Département d'informatique et recherche opérationnelle, Université de Montréal (2011)

[17] Rifai, S., Vincent, P., Muller, X., Glorot, X., Bengio, Y.: Contracting auto-encoders: Explicit invariance during feature extraction. In: Proceedings of the Twenty-eight International Conference on Machine Learning, ICML 2011 (2011)

[18] Salakhutdinov, R., Hinton, G.E.: Deep Boltzmann machines. In: AISTATS 2009, vol. 5, pp. 448–455 (2009)

[19] Simard, P., Victorri, B., LeCun, Y., Denker, J.: Tangent prop - A formalism for specifying selected invariances in an adaptive network. In: NIPS 1991, pp. 895–903. Morgan Kaufmann, San Francisco (1992)

[20] Swersky, K., Ranzato, M., Buchman, D., Marlin, B., de Freitas, N.: On score matching for energy based models: Generalizing autoencoders and simplifying deep learning. In: Proc. ICML 2011, ACM Press, New York (2011)
[21] Tikhonov, A.N., Arsenin, V.Y.: Solutions of Ill-posed Problems. W. H. Winston, Washington D.C (1977)
[22] Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.-A.: Extracting and composing robust features with denoising autoencoders. In: ICML 2008, pp. 1096–1103. ACM, New York (2008)
[23] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. JMLR 1, 3371–3408 (2010)
[24] Wahba, G.: Spline models for observational data. In: CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 59. SIAM, Philadelphia (1990)
[25] Weston, J., Ratle, F., Collobert, R.: Deep learning via semi-supervised embedding. In: Cohen, W.W., McCallum, A., Roweis, S.T. (eds.) ICML 2008, pp. 1168–1175. ACM, New York (2008)

The VC-Dimension of SQL Queries and Selectivity Estimation through Sampling*

Matteo Riondato**, Mert Akdere, Uğur Çetintemel, Stanley B. Zdonik, and Eli Upfal

Department of Computer Science, Brown University, Providence, RI, USA
{matteo,makdere,ugur,sbz,eli}@cs.brown.edu

Abstract. We develop a novel method, based on the statistical concept of VC-dimension, for evaluating the selectivity (output cardinality) of SQL queries – a crucial step in optimizing the execution of large scale database and data-mining operations. The major theoretical contribution of this work, which is of independent interest, is an explicit bound on the VC-dimension of a range space defined by all possible outcomes of a collection (class) of queries. We prove that the VC-dimension is a function of the maximum number of Boolean operations in the selection predicate, and of the maximum number of select and join operations in any individual query in the collection, but it is neither a function of the number of queries in the collection nor of the size of the database. We develop a method based on this result: given a class of queries, it constructs a concise random sample of a database, such that with high probability the execution of *any* query in the class on the sample provides an accurate estimate for the selectivity of the query on the original large database. The error probability holds *simultaneously* for the selectivity estimates of *all* queries in the collection, thus the same sample can be used to evaluate the selectivity of multiple queries, and the sample needs to be refreshed only following major changes in the database. The sample representation computed by our method is typically sufficiently small to be stored in main memory. We present extensive experimental results, validating our theoretical analysis and demonstrating the advantage of our technique when compared to complex selectivity estimation techniques used in PostgreSQL and the Microsoft SQL Server.

1 Introduction

As advances in technology allow for the collection and storage of vast databases, there is a growing need for advanced machine learning techniques for speeding up the execution of queries on such large datasets. In this work we focus on the fundamental task of estimating the selectivity, or output size, of a database query, which is a crucial step in a number of query processing tasks such as execution plan optimization and resource allocation in parallel and distributed databases.

* Work was supported in part by NSF award IIS-0905553.
** Corresponding author.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 661–676, 2011.

The task of efficiently obtaining such accurate estimates has been extensively studied with solutions ranging from storage of pre-computed statistics on the tables' distribution, to on-line sampling of the databases, and to combinations of the two approaches [39,40,24,30,25,15,16,20,31,37,49]. Histograms, simple yet powerful statistics of the tables' data, are the most commonly used solution in practice, due to their computational and space efficiency. However, there is an inherent limitation to the accuracy of this approach when estimating the selectivity of queries that involve either multiple tables/columns or correlated data. Running the query on freshly sampled data gives more accurate estimates at the cost of delaying the execution of the query while collecting random samples from a disk or other large storage medium and performing the analysis itself, which is usually more expensive than a histogram lookup. Our goal in this work is to leverage the computational efficiency of using pre-collected data with the provable accuracy of estimates obtained by running a query on a properly selected sample database.

We apply the statistical concept of VC-dimension [53] to develop and analyze a novel technique for generating accurate estimates of query selectivity. Roughly speaking, the VC-dimension of a collection of indicator functions (hypotheses) is a measure of its complexity or expressiveness (see Sect. 3 for formal definitions). A major theoretical contribution of this work, which is of independent interest, is an explicit bound to the VC-dimension of a class of queries, viewed as indicator functions on the Cartesian product of the database tables. In particular, we show that the VC-dimension of a class of queries is a function of the maximum number of Boolean, select and join operations in any query in the class, but it is not a function of the number of different queries in the class. By adapting a fundamental result from the VC-dimension theory to the database setting, we develop a method that for any class of queries, defined by its VC-dimension, constructs a concise sample of a database, such that with high probability, the execution of *any* query in the class on the sample provides an accurate estimate for the selectivity of the query on the original large database. The error probability holds *simultaneously* for the selectivity estimate of *all* queries in the collection, thus the same sample can be used to evaluate the selectivity of multiple queries, and the sample needs to be refreshed only following major changes in the database. The size of the sample does not depend on the size (number of tuples) in the database, just on the complexity of the class of queries we plan to run, measured by its VC-dimension. Both the analysis and the experimental results show that accurate selectivity estimates are obtained using a surprising small sample size (see Table 1 for concrete values), which allows the entire sample to reside in main memory, significantly speeding up the execution of the query on the sample.

A technical difficulty in applying the VC-dimension approach to the database setting is that the VC-dimension analysis assumes a uniform sample of the Cartesian products of all the tables, while in practice, it is more efficient to run the queries on the Cartesian product of random samples of the individual tables (which has a different distribution). We develop an efficient procedure for constructing a sample that circumvents this problem (see Sect. 5).

We present extensive experimental results, validating our theoretical analysis and demonstrating the advantage of our technique when compared to complex selectivity estimation techniques used in PostgreSQL and the Microsoft SQL Server. The main advantage of our methods is that it gives provably accurate prediction for all queries with up to a given complexity (VC-dimension), while techniques like multidimensional histograms or join synopses are accurate only for the queries for which they are built.

Note that we are only concerned with estimating the selectivity of a query, not with approximating the query answer using a sample of the database (see the work by Das [12] for a survey of that area).

Due to space limitation we focus here on the main novel concepts of our method. Full proofs and additional experimental results are included in the full paper [51].

2 Related Work

Methods for estimating the selectivity (or cardinality) of queries have been extensively studied in the database literature. A variety of approaches have been explored, ranging from the use of sampling, both online and offline, to precomputation of different statistics such as histograms, to building on methods from machine learning [11,28], data mining [21], optimization [7,42], and probabilistic modeling [19,50].

The use of sampling for selectivity estimation has been studied mainly in the context of online sampling [40,39], where the sample is obtained after a query has arrived and only used for the evaluation of the selectivity of that one query. Sampling at random from a large database residing on disk is an expensive operation [47,4,18], and in some cases sampling for an accurate cardinality estimate is not significantly faster than full execution of the query [22,23]. A variety of sampling and statistical analysis techniques has been tested for improving the efficiency of the sampling procedures and in particular identifying early stopping conditions [30,24,25,15,56,3,14,7,35] but online sampling is still considered too expensive for most applications. An offline sampling approach was explored by Ngu et al. [46] who used systematic sampling (requiring the tuples in a table to be sorted according to one of the attributes) with a sample size dependent on the number of tuples in the table. The paper does not give any explicit guarantee on the accuracy of their predictions. Chaudhuri et al. [7] present an approach which uses optimization techniques to identify suitable strata before sampling. The obtained sample is such that the mean square error in estimating the selectivity of queries belonging to a given workload is minimized, but there is no quality guarantee on the error for each of the queries. Haas [27] developed Hoeffding inequalities to bound the probability that the selectivity of a query estimated from a sample deviates more than a given amount from its expectation. However, to estimate the selectivity for multiple queries and obtain a given level accuracy for all of them, simultaneous statistical inference techniques like the union bound should be used, which are known to be overly conservative when the number of

queries is large [45]. On the contrary, our result will hold simultaneously for *all* queries within a given complexity (VC dimension).

A technical problem arises when combining join operations and sampling, as pointed out by Chaudhuri et al. [9]: the Cartesian product of the samples of two tables is not a uniform sample of the Cartesian product of the tables. What is more, given a size s, it is impossible a priori to determine two sample sizes s_1 and s_2 such that samples of these sizes from the two tables will give, when joined together along a common column, a sample of the join table of size s. In Sect. 5 we explain why only the first issue is of concern for us and how we circumvent it.

In practice most database systems use pre-computed statistics to predict query selectivity [31,20,16,34,37], with histograms being the most commonly used representation. The construction, maintenance, and use of histograms were thoroughly examined in the literature [33,32,43,48], with both theoretical and experimental results. In particular Chaudhuri et al. [8] rigorously evaluated the size of the sample needed for building a histogram providing good estimates for the selectivities of a large group of (select only, in their case) queries. Kaushik et al. [36] extensively compared histograms and sampling from a space complexity point of view, although their sample-based estimator did not offer a uniform probabilistic guarantee over a set of queries and they only consider the case of foreign-key equijoins. We address both these points in our work. Although very efficient in terms of storage needs and query time, the quality of estimates through histograms is inherently limited for complex queries by two major drawbacks: intra-bucket uniformity assumption (i.e., assuming a uniform distribution for the frequencies of values in the same bucket) and inter-column independence assumption (i.e., assuming no correlation between the values in different columns of the same or of different tables). Different authors suggested solutions to improve the estimation of selectivity without making the above assumptions [5,13,49,55,54]. Among these solutions, the use of multidimensional histograms [6,49,52,54] seems the most practical. Nevertheless, these techniques are not widespread due to the extra memory and computational costs in their implementation.

To the best of our knowledge, our work is the first to provide explicit bounds on the VC-dimension of queries and to apply the results to query selectivity estimation.

3 Preliminaries

We consider a database $\mathcal{D}$ of k tables $\mathcal{T}_1, \ldots, \mathcal{T}_k$. We denote a column C of a table $\mathcal{T}$ as $\mathcal{T}.C$ and, for a tuple $t \in \mathcal{T}$, the value of t in the column C as $t.C$. The values in $\mathcal{T}.C$ belong to the numerical or categorical domain $D(\mathcal{T}.C)$. Our focus is on queries that combine select and join operations, defined as follows. We do not take projection operations into consideration because their selectivities have no impact on query optimization.

Definition 1. *Given a table $\mathcal{T}$ with columns $\mathcal{T}.C_1, \ldots, \mathcal{T}.C_\ell$, a* selection query *$q$ on $\mathcal{T}$ is an operation which returns a subset S of the tuples of $\mathcal{T}$ such that a*

tuple t of $\mathcal{T}$ belongs to S if and only if the values in t satisfy a condition $\mathcal{C}$ (the selection predicate*) expressed by q.*

Definition 2. *Given two tables $\mathcal{T}_1$ and $\mathcal{T}_2$, a* join query *q on a common column C (i.e. a column present both in $\mathcal{T}_1$ and $\mathcal{T}_2$) is an operation which returns a subset of the Cartesian product of the tuples in $\mathcal{T}_1$ and $\mathcal{T}_2$. The returned subset is defined as the set $\{(t_1, t_2) \ : \ t_1 \in \mathcal{T}_1, t_2 \in \mathcal{T}_2, \ s.t. \ t_1.C \,\mathrm{op}\, t_2.C\}$ where "*op*" is one of $\{<, >, \geq, \leq, =, \neq\}$.*

Definition 3. *Given a set of ℓ tables $\mathcal{T}_1, \ldots, \mathcal{T}_\ell$, a* combination of select and join operations *is a query returning a subset of the Cartesian product of the tuples in the sets $S_1, \ldots, S_\ell$, where S_i is the output of a selection query on $\mathcal{T}_i$. The returned set is defined by the selection queries and by a set of join queries on $S_1, \ldots, S_\ell$.*

Definition 4. *Given a query q, a* query execution plan *for q is a directed binary tree T_q whose nodes are the elementary operations (i.e. select or join queries) into which q can be decomposed. There is an edge from node A to node B if the output of A is used as an input to B.*

It follows from the definition of a combination of select and join operations that a query may have multiple execution plans. Nevertheless, for all the queries we defined there is (at least) one execution plan such that all select operations are in the leaves and internal nodes are join nodes [17]. To derive our results, we use these specific plans.

Two crucial definitions that we use throughout the work are the *cardinality* of the output of a query and the equivalent concept of *selectivity* of a query.

Definition 5. *Given a query q and a database $\mathcal{D}$, the* cardinality *of its output is the number of elements (tuples if q is a selection queries, pairs of tuples if q is a join query, and ℓ-uples of tuples for combinations of join and select) in its output, when run on $\mathcal{D}$. The* selectivity *$\sigma(q)$ of q is the ratio between its cardinality and the product of the sizes of its input tables.*

VC-Dimension. The Vapnik-Chernovenkis (VC) Dimension of a space is a measure of complexity or expressiveness of a set of functions on that space [53]. A finite bound on the VC-dimension of a structure implies a bound on the size of random samples required for approximately learning that structure. We outline some basic definitions and results and their adaptation to the specific setting of queries. We refer the reader to the works of Alon and Spencer [2, Sect.14.4], and Chazelle [10, Chap. 4] for an in-depth discussion of the VC-dimension theory. VC-dimension is defined on *range spaces*:

Definition 6. *A* range space *is a pair (X, R) where X is a (finite or infinite) set and R is a (finite or infinite) family of subsets of X. The members of X are called* points *and those of R are called* ranges*.*

In our setting, for a class of select queries Q on a table $\mathcal{T}$, X is the set of all tuples in the input table, and R the family of the outputs of the queries in Q when run

on X, i.e. on $\mathcal{T}$. For a class Q of queries combining select and join operations, X is the Cartesian product of the associated tables and R is the family of outcomes of queries in Q, seen as ℓ-uples of tuples, if ℓ tables are involved in the queries of Q. When the context is clear we identify the family R with a class of queries.

Definition 7. *Let (X, R) be a range space and $A \subset X$. The* projection *of R on A is defined as $P_R(A) = \{r \cap A \ : \ r \in R\}$.*

Definition 8. *Let (X, R) be a range space and $A \subset X$. If $|P_R(A)| = 2^A$, then A is said to be* shattered by *R.*

Definition 9. *Let $S = (X, R)$ be a range space. The* Vapnik-Chervonenkis *dimension (or* VC-dimension*) of S, denoted as $VC(S)$ is the maximum cardinality of a shattered subset of X. If there are arbitrary large shattered subsets, then $VC(S) = \infty$.*

When the range space represents all the possible outputs of queries in Q applied to database tables $\mathcal{D}$, the VC-dimension of the range space is the maximum number of tuples such that any subset of them is defined by a query in Q.

The main application of VC-dimension in statistics and learning theory is its relation to the minimum sample size needed for approximate learning of a function on the point space using a range.

Definition 10. *Let (X, R) be a range space and let A be a finite subset of X.*

1. *For $0 < \varepsilon < 1$, a subset $B \subset A$ is an* ε-approximation *for A if for any range $r \in R$, we have $\left|\frac{|A \cap r|}{|A|} - \frac{|B \cap r|}{|B|}\right| \leq \varepsilon$.*
2. *For $0 < p, \varepsilon < 1$, a subset $B \subset A$ is a* relative (p, ε)-approximation *for A if for any range $r \in R$ such that $\frac{|A \cap r|}{|A|} \geq p$ we have $\left|\frac{|A \cap r|}{|A|} - \frac{|B \cap r|}{|B|}\right| \leq \varepsilon \frac{|A \cap r|}{|A|}$ and for any range $r \in R$ such that $\frac{|A \cap r|}{|A|} < p$ we have $\frac{|B \cap r|}{|B|} \leq (1 + \varepsilon)p$.*

It is possible to probabilistically build an ε-approximation (resp. a relative (p, ε)-approximation) by sampling the point space [53, 38, 29].

Theorem 1. *There is a positive constant c (resp. c') such that if (X, R) is a range-space of VC-dimension at most d, $A \subset X$ is a finite subset and $0 < \varepsilon, \delta < 1$ (resp. and $0 < p < 1$), then a random subset $B \subset A$ of cardinality m, where*

$$m \geq \min\left\{|A|, \frac{c}{\varepsilon^2}\left(d + \log\frac{1}{\delta}\right)\right\}, \tag{1}$$

(resp. $m \geq \min\left\{|A|, \frac{c'}{\varepsilon^2 p}\left(d \log\frac{1}{p} + \log\frac{1}{\delta}\right)\right\}$) is an ε-approximation (resp. a relative (p, ε)-approximation) for A with probability at least $1 - \delta$.

Löffler and Phillips [41] showed experimentally that the constant c is approximately 0.5. It is also interesting to note that an ε-approximation of size $O(\frac{d}{\varepsilon^2} \log\frac{d}{\varepsilon})$ can be built *deterministically* in time $O(d^{3d}(\frac{1}{\varepsilon^2} \log\frac{d}{\varepsilon})^d |X|)$ [10].

4 The VC-Dimension of Classes of Queries

In this section we develop a general bound on the VC-dimension of classes of queries. We start by computing the VC-dimension of simple select queries and move to queries with complex selection predicates and to join queries. We then extend our bounds to general queries that are combinations of multiple select and join operations. The proofs for our results can be found in the full version of this paper [51].

Select Queries. Let $\mathcal{T}$ be a table with m columns $\mathcal{T}.C_1, \ldots, \mathcal{T}.C_m$. For a fixed column $\mathcal{T}.C_i$, consider the family $\Sigma^*_{C_i}$ of all possible outputs of queries in the form

$$\texttt{SELECT} \ * \ \texttt{FROM} \ \mathcal{T} \ \texttt{WHERE} \ \mathcal{T}.C_i \,\mathrm{op}\, a \tag{2}$$

where op is an inequality operator (i.e., either "$\geq$" or "$\leq$")[1] and $a \in D(\mathcal{T}.C_i)$.

Lemma 1. *Let $\mathcal{T}$ be a table with m columns C_i, $1 \leq i \leq m$, and consider the set of queries $\Sigma^*_{\mathcal{T}} = \bigcup_{i=1}^{m} \Sigma^*_{C_i}$, where $\Sigma^*_{C_i}$ is defined as in the previous paragraph. Then, the range space $S = (\mathcal{T}, \Sigma^*_{\mathcal{T}})$ has VC-dimension at most $m + 1$.*

This lemma follows easily from a well known result on the VC-dimension of half-spaces in $\mathbb{R}^m$ [44, Lemma 10.3.1].

We now extend the bound to general selection queries with complex predicates. Given a table $\mathcal{T}$ with m columns, consider the set $\Sigma^{b*}_{\mathcal{T}}$ of queries whose selection predicate can be expressed as the set of selection queries whose predicate is a Boolean combination of at most b clauses, i.e. the query is in the form

$$\texttt{SELECT} \ * \ \texttt{FROM} \ \mathcal{T} \ \texttt{WHERE} \ \mathcal{T}.C_{i_1} \,\mathrm{op}_1\, a_1 \,\mathrm{bool}_1 \cdots \mathrm{bool}_{b-1}\, \mathcal{T}.C_{i_b} \,\mathrm{op}_b\, a_b$$

where op_i is one of "$\geq$", "$\leq$", "bool_ℓ" is either `AND` or `OR`, $i_j \in [1, m]$, and $a_j \in D(\mathcal{T}.C_{i_j})$, $1 \leq j \leq b$. It should be noted that the clauses in the selection predicate may be parenthesized in many different ways, each resulting (potentially) in a different query. All the possible parenthesizations are members of the range space $\Sigma^{b*}_{\mathcal{T}}$. It is also important to realize that we can and we do see a selection clause involving the "$=$" operator or the "$\neq$" operator as the "`AND`" of two clauses involving the $>$ and $<$ operators.

Lemma 2. *Let $\mathcal{T}$ be a table with m columns, let $b > 0$ and let $\Sigma^{b*}_{\mathcal{T}}$ be the set of selection queries on $\mathcal{T}$ whose selection predicate is a Boolean combination of up to b clauses. Then, the VC-dimension of the range space $S_b = (\mathcal{T}, \Sigma^{b*}_{\mathcal{T}})$ is at most $3((m+1)b)\log((m+1)b)$.*

Note that not all queries in $\Sigma^{b*}_{\mathcal{T}}$ are equivalent to axis-aligned boxes, thus we can not apply the bound used in the proof of Lemma 1. Instead, we use the following extension of [2, Corol. 14.4.3] to arbitrary combinations of set operations.

[1] The operators "$>$" and "$<$" can be reduced to "$\geq$" and "$\leq$" respectively.

Lemma 3. *Let (X, R) be a range space of VC-dimension $d \geq 2$ and let (X, R_h) be the range space on X in which R_h include all possible combinations of union and intersections of h members of R. Then $VC(X, R_h) \leq 3dh \log(dh)$.*

To prove Lemma 2 using Lemma 3, we observe that the outcome of an `AND` (resp. `OR`) operator connecting two selection clauses is equal to the intersection (resp. union) of the two selection clauses outputs.

Join Queries. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two distinct tables, and let R_1 and R_2 be families of (outputs of) queries on the tuples of $\mathcal{T}_1$ and $\mathcal{T}_2$ respectively. Let $S_1 = (\mathcal{T}_1, R_1)$, $S_2 = (\mathcal{T}_2, R_2)$ and let $VC(S_1), VC(S_2) \geq 2$. Let C be a column along which $\mathcal{T}_1$ and $\mathcal{T}_2$ are joined, and let $T_J = \mathcal{T}_1 \times \mathcal{T}_2$ be the Cartesian product of the two tables. For a pair of queries $r_1 \in R_1$, $r_2 \in R_2$, let

$$J^{\text{op}}_{r_1,r_2} = \{(t_1, t_2) \ : \ t_1 \in r_1, t_2 \in r_2, t_1.C \,\text{op}\, t_2.C\},$$

where op $\in \{>, <, \geq, \leq, =, \neq\}$. We have $J^{\text{op}}_{r_1,r_2} \subseteq r_1 \times r_2$ and $J^{\text{op}}_{r_1,r_2} \subseteq T_J$. Let $J_C = \{J^{\text{op}}_{r_1,r_2} \mid r_1 \in R_1, r_2, R_2, \text{op} \in \{>, <, \geq, \leq, =, \neq\}\}$. We have $VC((T_J, J_C)) \leq 3(VC(S_1) + VC(S_2)) \log((VC(S_1) + VC(S_2)))$. This result can be extended to queries with multiple joins operations:

Lemma 4. *Consider the class Q of queries that can be seen as combinations of select and join operations on $u > 2$ tables $\mathcal{T}_1, \ldots, \mathcal{T}_u$. Let $S_i = (\mathcal{T}_i, R_i)$, $i = 1, \ldots, u$ be the range space associated with the select queries on the u tables. Let $v_i = VC(S_i)$. Let m be the maximum number of columns in a table $\mathcal{T}_i$. We assume $m \leq \sum_i v_i$.*[2] *Let $S_Q = (\mathcal{T}_1 \times \cdots \times \mathcal{T}_u, R_Q)$ be the range space associated with the class Q. The range set R_Q is defined as follows. Let $\rho = (r_1, \ldots, r_u)$, $r_i \in R_i$, and let ω be a sequence of $u - 1$ join conditions representing a possible way to join the u tables $\mathcal{T}_i$, using the operators $\{>, <, \geq, \leq, =, \neq\}$. We define the range*

$$J^{\omega}_{\rho} = \{(t_1, \ldots, t_u) \ : \ t_i \in r_i, \ \textit{s.t.} \ (t_1, \ldots, t_u) \ \textit{satisfies} \ \omega\}.$$

R_Q is the set of all possible J^{ω}_{ρ}. Then,

$$VC(S_Q) \leq 4u(\sum_i VC(S_i)) \log(u \sum_i VC(S_i)).$$

We could not reduce the claim of this lemma to any know result in the VC-dimension theory. Our proof constructs a bound on the maximum size of a shattered set, by considering the effect of the different operators on the output. See the full version of the paper [51] for more details.

General Queries. Combining the above results we prove:

Theorem 2. *Consider the class $Q_{u,m,b}$ of all queries with up to $u - 1$ join and u select operations, where each select operation involves no more than m columns and b Boolean operations, then $VC(Q_{u,m,b}) \leq 12u^2(m + 1)b \log((m + 1)b) \log(3u^2(m + 1)b \log((m + 1)b))$.*

[2] The assumption $m \leq \sum_i v_i$ is reasonable for any practical case.

5 Implementation

Consider a database $\mathcal{D}$ and a class of queries $Q_{u,m,b}$. Theorem 2 gives a bound to the VC-dimension of the range space $(\mathbf{D}, Q_{u,m,b})$, where $\mathbf{D}$ is the Cartesian product of all the tables in $\mathcal{D}$. Theorem 1 gives the required size of a uniform random sample $\mathcal{S}$ of $\mathbf{D}$, such that the execution of *any* query $q \in Q_{u,m,b}$ on $\mathcal{S}$ gives an ε-approximation (or a relative (p, ε)-approximation) of the selectivity of q when executed on $\mathcal{D}$ (see Table 1 for concrete values). Note that for any execution plan of a query $q \in Q_{u,m,b}$, all the queries that correspond to subtrees rooted at internal nodes of the plan are queries in $Q_{u,m,b}$. Thus, by running query q on the sample we obtain accurate estimates for the selectivity of all the subqueries defined by its execution plan.

In practice, it is more efficient to maintain the table structure of the original database in the sample. It is easier to sample each table independently, and to run the query on a sample that consists of subsets of the original tables rather than re-writing the query to run on a Cartesian product of tuples. However, the Cartesian product of independent uniform samples of tables is not a uniform sample of the Cartesian product of the tables [9]. We developed a procedure to circumvent this problem. Due to space constraints, we present here an informal description of the procedure and refer the interested reader to the full version of the paper [51]. Assume that we need a uniform sample of size t from $\mathbf{D}$, which is the Cartesian product of ℓ tables $\mathcal{T}_1, \ldots, \mathcal{T}_\ell$. We then sample t tuples uniformly at random from each table $\mathcal{T}_i$, to form a sample table $\mathcal{S}_i$. We add an attribute *sampleindex* to each $\mathcal{S}_i$ and we set the value in the added attribute for each tuple in $\mathcal{S}_i$ to a unique value in $[1, t]$. Now, each sample table will contain t tuples, each tuple with a different index value in $[1, t]$. Given an index value $i \in [1, t]$, consider the set of tuples $X_i = \{x_1, \ldots, x_\ell\}$, $x_j \in \mathcal{S}_i$ such that $x_1.sampleindex = x_2.sampleindex = \cdots = x_\ell.sampleindex = i$. X_i can be seen as a tuple sampled from $\mathbf{D}$, and the set of all X_i, $i \in [1, t]$ is a uniform random sample of size t from $\mathbf{D}$. We run queries on the sample tables, but in order to estimate the selectivity of a join operation we count a tuple Y in the result only if the set of tuples composing Y is a subset of X_i for some $i \in [1, t]$. This is easily done by scanning the results and checking the values in the *sampleindex* columns.

Note that our method circumvent the major difficulty pointed out in [9]. They proved that, in general, it is impossible to predict sample sizes for given two tables such that the join of the samples of two tables will result in a sample of a required size out of the join of the two tables. Our method does not require a sample of a given size from the result of a join. The VC-dimension sampling technique requires only a sample of a given size from the Cartesian product of the tables, which is guaranteed by the above procedure.

6 Experiments

The first goal of the experiments is to evaluate the practical usefulness of our theoretical results. To assess this, we run queries on a large database and on

Table 1. Sample Sizes (tuples)

		Select		Join	
m	b	VC-dim	Sample size	VC-dim	Sample size
1	1	2	1000	4	1400
	2	4	1400	16	3800
	3	6	2800	36	7800
	5	10	2600	100	20600
	8	16	3800	256	51800
2	2	31	6800		
	3	57	12000		
	5	117	24000		
	8	220	44600		
5	5	294	59400		

sample representations generated by our method.[3] We used the selectivity of the each query in the random samples as an estimator for the selectivity in the large database (with the appropriate adjustments for join operations, as described in the previous section). We computed the error between the estimate and the actual selectivity to validate the analysis in Thm. 1 in practical settings. The use of a large number of queries and a variety of parameters allowed us to evaluate the error a function of the sample size. Our second goal is comparing our results, which give probabilistic guarantees on the error of the predicted selectivity, with the standard selectivity estimation done using precomputed statistics as implemented in PostgreSQL and Microsoft SQL Server. Additional experimental results can be found in the full version of the paper [51].

Setup. The tables in our large database were randomly generated and contain 20 million tuples. The distributions of values in the columns fell in two different categories: **1.** *uniform and independent*: the values in the columns were chosen uniformly and independently at random from a fixed domain. Each column was treated independently from the others. Tables involved in join queries belonged to this category only. **2.** *correlated*: two columns of the tables contained values from a multivariate normal distribution with mean $M = \mu\mathbb{I}_{2,2}$ and a non-identity covariance matrix Σ. We sampled tuples from the large tables uniformly, independently, and with replacement, to build the sample tables. For the samples of the tables used to run join queries, we added a column *sampleindex* to each tuple as described in Sect. 5. For each table in the original database we created many sample tables of different sizes, either fixed arbitrarily to 1000, 2000, or 5000 tuples or computed using (1). To compute the VC-dimension-dependent sample size, we fixed $\varepsilon = 0.05$, $\delta = 0.05$, and $c = 0.5$, as suggested by Löffler and Phillips [41]. The parameter d was set to the best bound to the VC-dimension of the range space of the queries we were running, as obtained from our theoretical results. We used $m = 1, 2$ (only $m = 1$ for joins) and $b = 1, 2, 3, 5, 8$, with the

[3] We focused on building ε-approximations, but relative (p, e)-approximations would give similar results.

addition of the combination $m = 5$, $b = 5$. Table 1 shows the sample sizes as number of tuples. We stress again the fact that the sample sizes are independent from the sizes of the original tables, so the larger the original table, the smaller will be the ratio between the sample size and the original size and the higher the gains in terms of space. We built PostgreSQL histograms with a different number of buckets, ranging from 100 to 10000. For SQL Server, we built the standard single-column histograms and computed the multi-column statistics which should help obtaining better estimations when the values along the columns are correlated. For each combination of the parameters m and b and each large table (or pair of large tables, in the case of join) we created 100 queries, with selection predicates chosen independently and uniformly at random, involving m columns and b Boolean clauses.

Results. A major result of our experiments is that for all the queries we run and all the sample tables the estimate of the selectivity computed using the selectivity in the sample was within ε (0.05) from the real selectivity. The same was not true for the selectivity computed by the histograms. As an example, in the case of $m = 2$, $b = 5$ and uniform independent columns, the default PostgreSQL histograms predicted a selectivity more than ε off from the real selectivity for 30 out of 100 queries. Nevertheless, from time to time the histograms predicted a selectivity closer to the actual one than the prediction from the sample. This is especially true when the histogram assumption are verified (e.g., for $m = 1, b = 1$ the default PostgreSQL histograms gave a better prediction than the sample in 28 out of 100 cases). This "inversion of precision" becomes less and less frequent as the sample size grows and as the complexity of the queries grows. Since the selectivity estimated by the sample was *always* within ε from the actual, we focused on the percent error, i.e. on the quantity $e_{\%} = \frac{100|p(\sigma_q) - \sigma_{\mathcal{D}}(q)|}{\sigma_{\mathcal{D}}(q)}$ where $p(\sigma_q)$ is the predicted selectivity. We can see from Fig. 1 and 2 that both the average and the standard deviation of the percentage error of the sample prediction decrease as the sample size grows. Much more interesting than this is the comparison between the performance of the histograms and the performance of the sample in predicting selectivities. When the assumptions of the histograms hold, as is the case for the data plotted in Fig. 1, the predictions obtained from the histograms can be of good quality. As said though, for a majority of queries, the prediction from the sample is better than the one from the histograms.

But as soon as the data are correlated (Fig. 2), the sample gives better predictions than the histograms even at the smallest sample sizes and keeps improving as the sample grows larger. In Fig. 2 we do not show multiple curves for the different PostgreSQL histograms because increasing the number of buckets had very marginal impact on the quality of the estimates, sometime even in the negative sense (i.e., an histogram with more buckets gave worse predictions than an histogram with less buckets), a fact that can be explained with the variance introduced by the sampling process used to create the histograms. For the same reason we do not plot multiple lines for the prediction obtained from the multi-columns and single-column statistics of SQL Server. The strengths of our method compared to histograms become more evident when we run join queries.

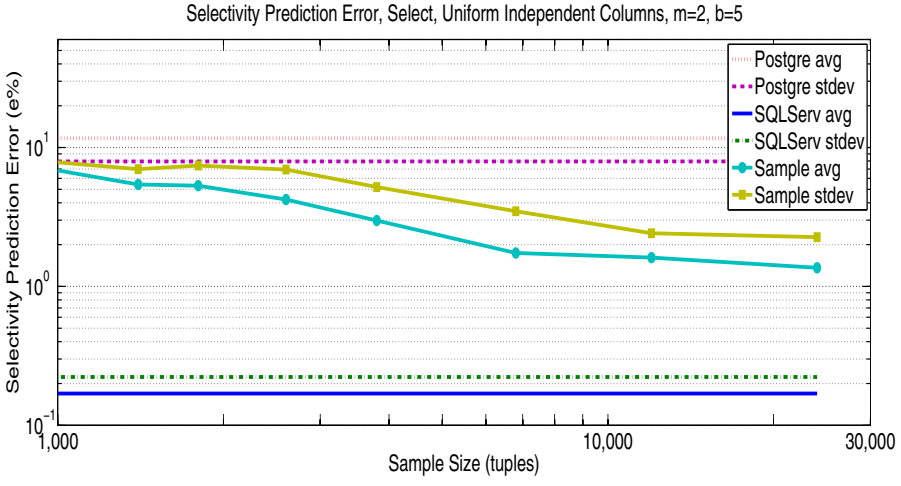


Fig. 1. Select – Uniform Independent Columns – $m = 2$, $b = 5$

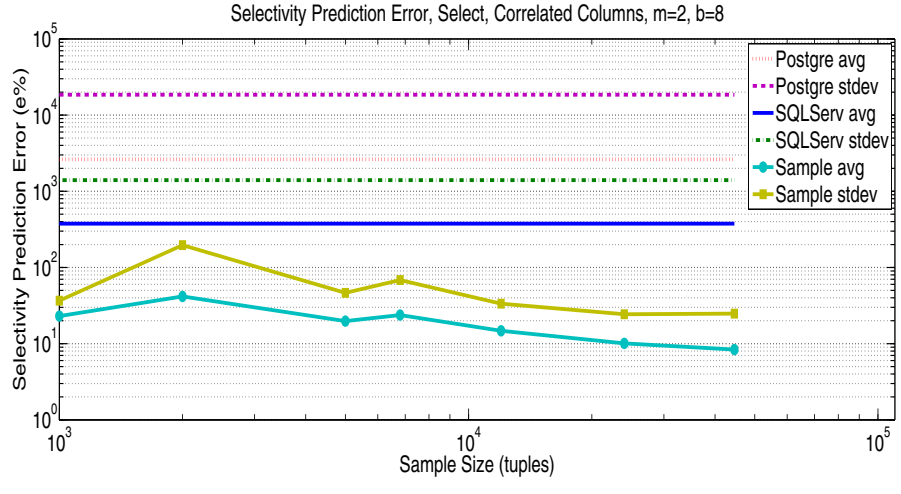


Fig. 2. Select – Correlated Columns – $m = 2$, $b = 8$

In our experiments, the predictions obtained from the sample were always within ε from the real values, even at the smallest sample sizes, but the same was not true for histograms. Figure 3 shows the comparison between the average and the standard deviation of the percentage error for the histograms and the sample. The numbers include predictions for the selection operations at the leaves of the query tree. The extremely bad performances of PostgreSQL is due to the fact that for some join queries, the histograms may predict an output size on the order of the hundreds of thousands tuples but the actual output size was zero or a very small number of tuples. Such errors drive the average and the standard

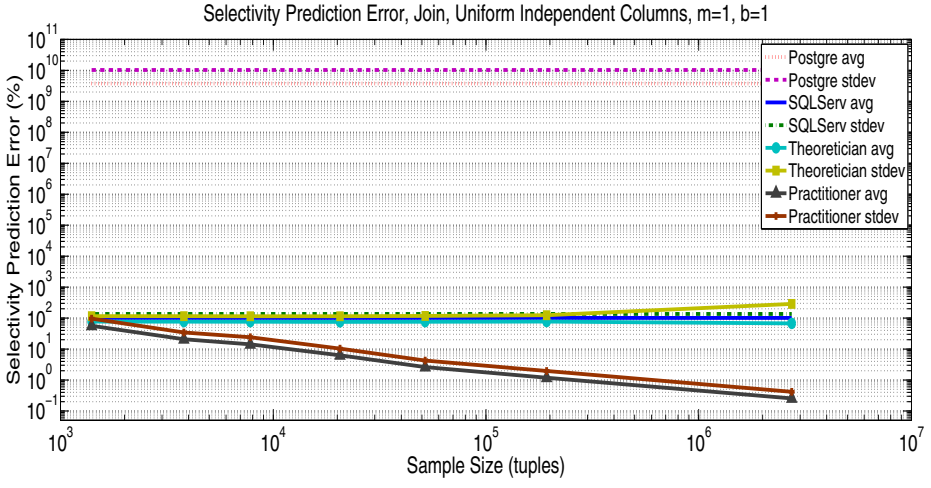


Fig. 3. Join – Uniform Independent Columns – $m = 1$, $b = 1$

deviation to very high values, but the comparison with the sample is fair and the prediction from the histograms are just not of good quality in such cases. The performance of SQL Server can be explained by the fact that this DBMS does not only use histograms to predict the selectivity of the query but also analyzes the query predicates and its query optimizer is very good in understanding when a query would return an empty output, therefore avoiding major errors in the estimation. This is something that vanilla histograms could not do, so the comparison with the sample is actually a bit unfair against the sample. Figure 3 also shows a comparison between the percentage error of predictions obtained from the sample in two different ways: the "theoretically correct" way that makes use of the number of pairs of tuples with the same value in the *sampleindex* column to predict the selectivity and the "practitioner" way which uses the size of the output of the join operation in the sample, ignoring the *sampleindex* column, i.e., without filtering out the tuples not belonging to the sample of the Cartesian product of the original tables.

From the experiments we ran we can conclude that our method for estimating the selectivity is a viable option in practice. The theoretical guarantees were always satisfied, with a consistency and an accuracy even higher than guaranteed. This fact can be explained by the potential looseness of the bounds to the VC-dimension of queries, and therefore to the sample size. From a practical point of view, it is also interesting that a sample of the Cartesian product is not necessary, and a certain level of non-uniformity in the sampling process may be accommodated. It may even well be that it is not necessary to use uniform independent sampling in order to obtain a ε-approximation.

7 Conclusions

We develop a novel method for estimating the selectivity of queries by executing it on a concise, properly selected, sample of the database. We present a rigorous analysis of our method and extensive experimental results demonstrating its efficiency and the accuracy of its predictions.

Most commercial databases use histograms built on a single column, for selectivity estimation. There has also been significant research on improving the estimate using multidimensional histograms [6,49,52,54] and join synopses [1]. The main advantage of our method is that it gives uniformly accurate estimate for the selectivity of any query within a predefined VC-dimension range. Method that collect and store pre-computed statistics gives accurate estimates only for the queries captured by the collected statistics, while estimates of any other query relies on an independence assumption.

To match the accuracy of our new method with histograms and join synopses one would need to create, for each table, a multidimensional histogram where the number of dimensions is equal to the number of columns in the tables. The space needed for a multidimensional histogram is exponential in the number of dimensions, while the size of our sample representation is almost linear in that parameter. Furthermore, to estimate the selectivity for join operations one would need to create join synopses for all pairs of columns in the database, again in space that grows exponential in the number of columns.

It is interesting to note that the highly theoretical concept of VC-dimension leads in this work to an efficient and practical tool for an important data analysis problem.

References

1. Acharya, S., Gibbons, P.B., Poosala, V., Ramaswamy, S.: Join Synopses for Approximate Query Answering. In: SIGMOD 1999 (1999)
2. Alon, N., Spencer, J.H.: The Probabilistic Method, 3rd edn. John Wiley & Sons, Chichester (2008)
3. Babcock, B., Chaudhuri, S., Das, G.: Dynamic Sample Selection for Approximate Query Processing. In: SIGMOD 2003 (2003)
4. Brown, P.G., Haas, P.J.: Techniques for Warehousing of Sample Data. In: ICDE 2006 (2006)
5. Bruno, N., Chaudhuri, S.: Conditional Selectivity for Statistics on Query Expressions. In: SIGMOD 2004 (2004)
6. Bruno, N., Chaudhuri, S., Gravano, L.: STHoles: a Multidimensional Workload-Aware Histogram. In: SIGMOD 2001 (2001)
7. Chaudhuri, S., Das, G., Narasayya, V.: Optimized Stratified Sampling for Approximate Query Processing. ACM Trans. Database Syst. 32(2), Art. 9 (2007)
8. Chaudhuri, S., Motwani, R., Narasayya, V.: Random Sampling for Histogram Construction: How Much is Enough. In: SIGMOD 1998 (1998)
9. Chaudhuri, S., Motwani, R., Narasayya, V.: On Random Sampling over Joins. In: SIGMOD 1999 (1999)

10. Chazelle, B.: The Discrepancy Method: Randomness and Complexity. Cambridge University Press, Cambridge (2000)
11. Chen, M.C., McNamee, L., Matloff, N.S.: Selectivity Estimation Using Homogeneity Measurement. In: ICDE 1990 (1990)
12. Das, G.: Sampling Methods in Approximate Query Answering Systems. In: Wang, J. (ed.) Encyclopedia of Data Warehousing and Mining. Information Science Publishing (2005)
13. Dobra, A.: Histograms Revisited: When are Histograms the Best Approximation Method for Aggregates over Joins? In: PODS 2005 (2005)
14. Estan, C., Naughton, J.F.: End-Biased Samples for Join Cardinality Estimation. In: ICDE 2006 (2006)
15. Ganguly, S., Gibbons, P.B., Matias, Y., Silberschatz, A.: Bifocal Sampling for Skew-Resistant Join Size Estimation. In: SIGMOD 1996 (1996)
16. Ganti, V., Lee, M.-L., Ramakrishnan, R.: ICICLES: Self-Tuning Samples for Approximate Query Answering. In: VLDB 2000 (2000)
17. Garcia-Molina, H., Ullman, J.D., Widom, J.: Database Systems: The Complete Book. Prentice-Hall, Englewood Cliffs (2002)
18. Gemulla, R., Lehner, W., Haas, P.J.: A Dip in the Reservoir: Maintaining Sample Synopses of Evolving Datasets. In: VLDB 2006 (2006)
19. Getoor, L., Taskar, B., Koller, D.: Selectivity Estimation using Probabilistic Models. In: SIGMOD 2001 (2001)
20. Gibbons, P.B., Matias, Y.: New Sampling-Based Summary Statistics for Improving Approximate Query Answers. In: SIGMOD 1998 (1998)
21. Gryz, J., Liang, D.: Query Selectivity Estimation via Data Mining. In: IIS 2004 (2004)
22. Haas, P.J., Naughton, J.F., Seshadri, S., Swami, A.N.: Fixed-Precision Estimation of Join Selectivity. In: PODS 1993 (1993)
23. Haas, P.J., Naughton, J.F., Swami, A.N.: On the Relative Cost of Sampling for Join Selectivity Estimation. In: PODS 1994 (1994)
24. Haas, P.J., Swami, A.N.: Sequential Sampling Procedures for Query Size Estimation. In: SIGMOD 1992 (1992)
25. Haas, P.J., Swami, A.N.: Sampling-Based Selectivity Estimation for Joins Using Augmented Frequent Value Statistics. In: ICDE 1995 (1995)
26. Haas, P.J., Naughton, J.F., Seshadri, S., Swami, A.N.: Selectivity and Cost Estimation for Joins Based on Random Sampling. Jour. of Comp. and Sys. Sci. 52, 550–569 (1996)
27. Haas, P.J.: Hoeffding Inequalities for Join-Selectivity Estimation and Online Aggregation. IBM Research Report RJ 10040 (2000)
28. Harangsri, B., Shepherd, J., Ngu, A.H.H.: Query Size Estimation using Machine Learning. In: DASFAA 1997 (1997)
29. Har-Peled, S., Sharir, M.: Relative (p, ε)-Approximations in Geometry. Discrete & Computational Geometry 45(3), 462–496 (2011)
30. Hou, W.-C., Ozsoyoglu, G., Dogdu, E.: Error-Constraint Count Query Evaluation in Relational Databases. In: SIGMOD 1991 (1991)
31. Hou, W.-C., Ozsoyoglu, G., Taneja, B.K.: Statistical Estimators for Relational Algebra Expressions. In: PODS 1988 (1988)
32. Ioannidis, Y.E., Poosala, V.: Balancing Histogram Optimality and Practicality for Query Result Size Estimation. In: SIGMOD 1995 (1995)
33. Jagadish, H.V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K.C., Suel, T.: Optimal Histograms with Quality Guarantees. In: VLDB 1998 (1998)

34. Jin, R., Glimcher, L., Jermaine, C., Agrawal, G.: New Sampling-Based Estimators for OLAP Queries. In: ICDE 2006 (2006)
35. Joshi, S., Jermaine, C.: Robust Stratified Sampling Plans for Low Selectivity Queries. In: ICDE 2008 (2008)
36. Kaushik, R., Naughton, J.F., Ramakrishnan, R., Chakravarthy, V.T.: Synopses for Query Optimization: a Space-Complexity Perspective. ACM Trans. Database Syst. 30(4), 1102–1127 (2005)
37. Larson, P.-A., Lehner, W., Zhou, J., Zabback, P.: Cardinality Estimation Using Sample Views with Quality Assurance. In: SIGMOD 2007 (2007)
38. Li, Y., Long, P.M., Srinivasan, A.: Improved Bounds on the Sample Complexity of Learning. Jour. of Comp. and Sys. Sci. 62, 516–527 (2001)
39. Lipton, R.J., Naughton, J.F.: Query Size Estimation by Adaptive Sampling. J. Comput. Syst. Sci. 51(1), 18–25 (1995)
40. Lipton, R.J., Naughton, J.F., Schneider, D.A.: Practical Selectivity Estimation through Adaptive Sampling. In: SIGMOD 1990 (1990)
41. Löffler, M., Phillips, J.M.: Shape fitting on point sets with probability distributions. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 313–324. Springer, Heidelberg (2009)
42. Markl, V., Haas, M., Peter, J., Kutsch, Megiddo, N., Srivastava, U., Tran, T.M.: Consistent Selectivity Estimation via Maximum Entropy. The VLDB Journal 16(1), 55–76 (2007)
43. Matias, Y., Vitter, J.S., Wang, M.: Wavelet-Based Histograms for Selectivity Estimation. In: SIGMOD 1998 (1998)
44. Matoušek, J.: Lectures on Discrete Geometry. Springer, Heidelberg (2002)
45. Miller, R.J.: Simultaneous Statistical Inference, 2nd edn. Springer, Heidelberg (1981)
46. Ngu, A.H., Harangsri, B., Shepherd, J.: Query Size Estimation for Joins Using Systematic Sampling. Distributed and Parallel Databases 15, 237–275 (2004)
47. Olken, F.: Random Sampling from Databases. Ph.D. dissertation, LBL Tech. Report LBL-32883 (1993)
48. Poosala, V., Haas, P.J., Ioannidis, Y.E., Shekita, E.J.: Improved Histograms for Selectivity Estimation of Range Predicates. In: SIGMOD 1996 (1996)
49. Poosala, V., Ioannidis, Y.E.: Selectivity Estimation without the Attribute Value Independence Assumption. In: VLDB 1997 (1997)
50. Ré, C., Suciu, D.: Understanding Cardinality Estimation Using Entropy Maximization. In: PODS 2010 (2010)
51. Riondato, M., Akdere, M., Çetintemel, U., Zdonik, S.B., Upfal, E.: The VC-Dimension of SQL Queries and Selectivity Estimation Through Sampling. CoRR abs/1101.5805 (2011)
52. Srivastava, U., Haas, P.J., Markl, V., Kutsch, M., Tran, T.: ISOMER: Consistent Histogram Construction Using Query Feedback. In: ICDE 2006 (2006)
53. Vapnik, V.N., Chervonenkis, A.Y.: On the Uniform Convergence of Relative Frequencies of Events to their Probabilities. Theory of Probability and its Applications 16(2), 264–280 (1971)
54. Wang, H., Sevcik, K.C.: A Multi-Dimensional Histogram for Selectivity Estimation and Fast Approximate Query Answering. In: CASCON 2003 (2003)
55. Wang, M., Vitter, J.S., Iyer, B.R.: Selectivity Estimation in the Presence of Alphanumeric Correlations.In: ICDE 1997 (1997)
56. Wu, Y.-L., Agrawal, D., El Abbadi, A.: Applying the Golden Rule of Sampling for Query Estimation. In: SIGMOD 2001 (2001)

Author Index