Joshua Zhexue Huang
Longbing Cao
Jaideep Srivastava (Eds.)

LNAI 6634

Advances in Knowledge Discovery and Data Mining

15th Pacific-Asia Conference, PAKDD 2011
Shenzhen, China, May 2011
Proceedings, Part I

Springer

Lecture Notes in Artificial Intelligence 6634

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Joshua Zhexue Huang Longbing Cao
Jaideep Srivastava (Eds.)

Advances in Knowledge Discovery and Data Mining

15th Pacific-Asia Conference, PAKDD 2011
Shenzhen, China, May 24-27, 2011
Proceedings, Part I

Springer

Volume Editors

Joshua Zhexue Huang
Chinese Academy of Sciences
Shenzhen Institutes of Advanced Technology (SIAT)
Shenzhen 518055, China
E-mail: zx.huang@siat.ac.cn

Longbing Cao
University of Technology Sydney
Faculty of Engineering and Information Technology
Advanced Analytics Institute
Center for Quantum Computation and Intelligent Systems
Sydney, NSW 2007, Australia
E-mail: longbing.cao-1@uts.edu.au

Jaideep Srivastava
University of Minnesota
Department of Computer Science and Engineering
Minneapolis, MN 55455, USA
E-mail: Srivasta@cs.umn.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-20840-9 e-ISBN 978-3-642-20841-6
DOI 10.1007/978-3-642-20841-6
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011926132

CR Subject Classification (1998): I.2, H.3, H.4, H.2.8, I.4, C.2

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)

Preface

PAKDD has been recognized as a major international conference in the areas of data mining (DM) and knowledge discovery in databases (KDD). It provides an international forum for researchers and industry practitioners to share their new ideas, original research results and practical development experiences from all KDD-related areas including data mining, machine learning, artificial intelligence and pattern recognition, data warehousing and databases, statistics, knowledge engineering, behavioral sciences, visualization, and emerging areas such as social network analysis.

The 15^{th} Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2011) was held in Shenzhen, China, during May 24–27, 2011. PAKDD 2011 introduced a double-blind review process. It received 331 submissions after checking for validity. Submissions were from 45 countries and regions, which shows a significant improvement in internationalization than PAKDD 2010 (34 countries and regions). All papers were assigned to at least four Program Committee members. Most papers received more than three review reports. As a result of the deliberation process, only 90 papers were accepted, with 32 papers (9.7%) for long presentation and 58 (17.5%) for short presentation.

The PAKDD 2011 conference program also included five workshops: the Workshop on Behavior Informatics (BI 2011), Workshop on Advances and Issues in Traditional Chinese Medicine Clinical Data Mining (AI-TCM), Quality Issues, Measures of Interestingness and Evaluation of Data Mining Models (QIMIE 2011), Biologically Inspired Techniques for Data Mining (BDM 2011), and Workshop on Data Mining for Healthcare Management (DMHM 2011). PAKDD 2011 also featured talks by three distinguished invited speakers, six tutorials, and a Doctoral Symposium on Data Mining.

The conference would not have been successful without the support of the Program Committee members (203), external reviewers (168), Organizing Committee members, invited speakers, authors, tutorial presenters, workshop organizers, reviewers, authors and the conference attendees. We highly appreciate the conscientious reviews provided by the Program Committee members and external reviewers. We are indebted to the members of the PAKDD Steering Committee for their invaluable suggestions and support throughout the organization process. Our special thanks go to the local arrangements team and volunteers. We would also like to thank all those who contributed to the success of PAKDD 2011 but whose names cannot be listed.

We greatly appreciate Springer LNCS for continuing to publish the main conference and workshop proceedings. Thanks also to Andrei Voronkov for hosting the entire PAKDD reviewing process on the EasyChair.org site.

Finally, we greatly appreciate the support from various sponsors and institutions. The conference was organized by the Shenzhen Institutes of Advanced

Technology, Chinese Academy of Sciences, China, and co-organized by the University of Hong Kong, China and the University of Technology Sydney, Australia.

We hope you enjoy the proceedings of PAKDD 2011, which presents cutting-edge research in data mining and knowledge discovery. We also hope all participants took this opportunity to exchange ideas with each other and enjoyed the modem city of Shenzhen!

May 2011

Joshua Huang
Longbing Cao
Jaideep Srivastava

Organization

Organizing Committee

Honorary Chair

Philip S. Yu	University of Illinois at Chicago, USA

General Co-chairs

Jianping Fan	Shenzhen Institutes of Advanced Technology, CAS, China
David Cheung	University of Hong Kong, China

Program Committee Co-chairs

Joshua Huang	Shenzhen Institutes of Advanced Technology, CAS, China
Longbing Cao	University of Technology Sydney, Australia
Jaideep Srivastava	University of Minnesota, USA

Workshop Co-chairs

James Bailey	The University of Melbourne, Australia
Yun Sing Koh	The University of Auckland, New Zealand

Tutorial Co-chairs

Xiong Hui	Rutgers, the State University of New Jersey, USA
Sanjay Chawla	The University of Sydney, Australia

Local Arrangements Co-chairs

Shengzhong Feng	Shenzhen Institutes of Advanced Technology, CAS, China
Jun Luo	Shenzhen Institutes of Advanced Technology, CAS, China

Sponsorship Co-chairs

Yalei Bi	Shenzhen Institutes of Advanced Technology, CAS, China
Zhong Ming	Shenzhen University, China

Publicity Co-chairs

Jian Yang	Beijing University of Technology, China
Ye Li	Shenzhen Institutes of Advanced Technology, CAS, China
Yuming Ou	University of Technology Sydney, Australia

Publication Chair

Longbing Cao	University of Technology Sydney, Australia

Steering Committee

Co-chairs

Rao Kotagiri	University of Melbourne, Australia
Graham Williams	Australian National University, Australia

Life Members

David Cheung	University of Hong Kong, China
Masaru Kitsuregawa	Tokyo University, Japan
Rao Kotagiri	University of Melbourne, Australia
Hiroshi Motoda	AFOSR/AOARD and Osaka University, Japan
Graham Williams (Treasurer)	Australian National University, Australia
Ning Zhong	Maebashi Institute of Technology, Japan

Members

Ming-Syan Chen	National Taiwan University, Taiwan, ROC
Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Ee-Peng Lim	Singapore Management University, Singapore
Huan Liu	Arizona State University, USA
Jaideep Srivastava	University of Minnesota, USA
Takashi Washio	Institute of Scientific and Industrial Research, Osaka University, Japan
Thanaruk Theeramunkong	Thammasat University, Thailand
Kyu-Young Whang	Korea Advanced Institute of Science and Technology, Korea
Chengqi Zhang	University of Technology Sydney, Australia
Zhi-Hua Zhou	Nanjing University, China
Krishna Reddy	IIIT, Hyderabad, India

Program Committee

Adrian Pearce	The University of Melbourne, Australia
Aijun An	York University, Canada
Aixin Sun	Nanyang Technological University, Singapore
Akihiro Inokuchi	Osaka University, Japan

Akira Shimazu	Japan Advanced Institute of Science and Technology, Japan
Alfredo Cuzzocrea	University of Calabria, Italy
Andrzej Skowron	Warsaw University, Poland
Anirban Mondal	IIIT Delhi, India
Aoying Zhou	Fudan University, China
Arbee Chen	National Chengchi University, Taiwan, ROC
Aristides Gionis	Yahoo Research Labs, Spain
Atsuyoshi Nakamura	Hokkaido University, Japan
Bart Goethals	University of Antwerp, Belgium
Bernhard Pfahringer	University of Waikato, New Zealand
Bo Liu	University of Technology, Sydney, Australia
Bo Zhang	Tsinghua University, China
Boonserm Kijsirikul	Chulalongkorn University, Thailand
Bruno Cremilleux	Université de Caen, France
Chandan Reddy	Wayne State University, USA
Chang-Tien Lu	Virginia Tech, USA
Chaveevan Pechsiri	Dhurakijpundit University, Thailand
Chengqi Zhang	University of Technology, Australia
Chih-Jen Lin	National Taiwan University, Taiwan, ROC
Choochart Haruechaiyasak	NECTEC, Thailand
Chotirat Ann Ratanamahatana	Chulalongkorn University, Thailand
Chun-hung Li	Hong Kong Baptist University, Hong Kong, China
Chunsheng Yang	NRC Institute for Information Technology, Canada
Clement Yu	University of Illinois at Chicago, USA
Dacheng Tao	The Hong Kong Polytechnic University, Hongkong, China
Daisuke Ikeda	Kyushu University, Japan
Dan Luo	University of Technology, Sydney, Australia
Daoqiang Zhang	Nanjing University of Aeronautics and Astronautics, China
Dao-Qing Dai	Sun Yat-Sen University, China
David Albrecht	Monash University, Australia
David Taniar	Monash University, Australia
Di Wu	Chinese University of Hong Kong, China
Diane Cook	Washington State University, USA
Dit-Yan Yeung	Hong Kong University of Science and Technology, China
Dragan Gamberger	Rudjer Boskovic Institute, Croatia
Du Zhang	California State University, USA
Ee-Peng Lim	Nanyang Technological University, Singapore
Eibe Frank	University of Waikato, New Zealand
Evaggelia Pitoura	University of Ioannina, Greece

Floriana Esposito	Università di Bari, Italy
Gang Li	Deakin University, Australia
George Karypis	University of Minnesota, USA
Graham Williams	Australian Taxation Office, Australia
Guozhu Dong	Wright State University, USA
Hai Wang	University of Aston, UK
Hanzi Wang	University of Adelaide, Australia
Harry Zhang	University of New Brunswick, Canada
Hideo Bannai	Kyushu University, Japan
Hiroshi Nakagawa	University of Tokyo, Japan
Hiroyuki Kawano	Nanzan University, Japan
Hiroyuki Kitagawa	University of Tsukuba, Japan
Hua Lu	Aalborg University, Denmark
Huan Liu	Arizona State University, USA
Hui Wang	University of Ulster, UK
Hui Xiong	Rutgers University, USA
Hui Yang	San Francisco State University, USA
Huiping Cao	New Mexico State University, USA
Irena Koprinska	University of Sydney, Australia
Ivor Tsang	Hong Kong University of Science and Technology, China
James Kwok	Hong Kong University of Science and Technology, China
Jason Wang	New Jersey Science and Technology University, USA
Jean-Marc Petit	INSA Lyon, France
Jeffrey Ullman	Stanford University, USA
Jiacai Zhang	Beijing Normal University, China
Jialie Shen	Singapore Management University, Singapore
Jian Yin	Sun Yat-Sen University, China
Jiawei Han	University of Illinois at Urbana-Champaign, USA
Jiuyong Li	University of South Australia
Joao Gama	University of Porto, Portugal
Jun Luo	Chinese Academy of Sciences, China
Junbin Gao	Charles Sturt University, Australia
Junping Zhang	Fudan University, China
K. Selcuk Candan	Arizona State University, USA
Kaiq huang	Chinese Academy of Sciences, China
Kennichi Yoshida	Tsukuba University, Japan
Kitsana Waiyamai	Kasetsart University, Thailand
Kouzou Ohara	Osaka University, Japan
Liang Wang	The University of Melbourne, Australia
Ling Chen	University of Technology Sydney, Australia
Lisa Hellerstein	Polytechnic Institute of NYU, USA

Longbing Cao	University of Technology Sydney, Australia
Manabu Okumura	Tokyo Institute of Technology, Japan
Marco Maggini	University of Siena, Italy
Marut Buranarach	NECTEC, Thailand
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Masashi Shimbo	Nara Institute of Science and Technology, Japan
Masayuki Numao	Osaka University, Japan
Maurice van Keulen	University of Twente, The Netherlands
Xiaofeng Meng	Renmin University of China, China
Mengjie Zhang	Victoria University of Wellington, New Zealand
Michael Berthold	University of Konstanz, Germany
Michael Katehakis	Rutgers Business School, USA
Michalis Vazirgiannis	Athens University of Economics and Business, Greece
Min Yao	Zhejiang University, China
Mingchun Wang	Tianjin University of Technology and Education, China
Mingli Song	Hong Kong Polytechnical University, China
Mohamed Mokbel	University of Minnesota, USA
Naren Ramakrishnan	Virginia Tech, USA
Ngoc Thanh Nguyen	Wroclaw University of Technology, Poland
Ning Zhong	Maebashi Institute of Technology, Japan
Ninghui Li	Purdue University, USA
Olivier de Vel	DSTO, Australia
Pabitra Mitra	Indian Institute of Technology Kharagpur, India
Panagiotis Karras	University of Zurich, Switzerland
Pang-Ning Tan	Michigan State University, USA
Patricia Riddle	University of Auckland, New Zealand
Panagiotis Karras	National University of Singapore, Singapore
Jialie Shen	Singapore Management University, Singapore
Pang-Ning Tan	Michigan State University, USA
Patricia Riddle	University of Auckland, New Zealand
Peter Christen	Australian National University, Australia
Peter Triantafillou	University of Patras, Greece
Philip Yu	IBM T.J. Watson Research Center, USA
Philippe Lenca	Telecom Bretagne, France
Pohsiang Tsai	National Formosa University, Taiwan, ROC
Prasanna Desikan	University of Minnesota, USA
Qingshan Liu	Chinese Academy of Sciences, China
Rao Kotagiri	The University of Melbourne, Australia
Richi Nayak	Queensland University of Technology, Australia
Rui Camacho	LIACC/FEUP University of Porto, Portugal
Ruoming Jin	Kent State University, USA

S.K. Gupta	Indian Institute of Technology, India
Salvatore Orlando	University of Venice, Italy
Sameep Mehta	IBM, India Research Labs, India
Sanjay Chawla	University of Sydney, Australia
Sanjay Jain	National University of Singapore, Singapore
Sanjay Ranka	University of Florida, USA
San-Yih Hwang	National Sun Yat-Sen University, Taiwan, ROC
Seiji Yamada	National Institute of Informatics, Japan
Sheng Zhong	State University of New York at Buffalo, USA
Shichao Zhang	University of Technology at Sydney, Australia
Shiguang Shan	Digital Media Research Center, ICT
Shoji Hirano	Shimane University, Japan
Shu-Ching Chen	Florida International University, USA
Shuigeng Zhou	Fudan University, China
Songcan Chen	Nanjing University of Aeronautics and Astronautics, China
Srikanta Tirthapura	Iowa State University, USA
Stefan Rueping	Fraunhofer IAIS, Germany
Suman Nath	Networked Embedded Computing Group, Microsoft Research
Sung Ho Ha	Kyungpook National University, Korea
Sungzoon Cho	Seoul National University, Korea
Szymon Jaroszewicz	Technical University of Szczecin, Poland
Tadashi Nomoto	National Institute of Japanese Literature, Tokyo, Japan
Taizhong Hu	University of Science and Technology of China
Takashi Washio	Osaka University, Japan
Takeaki Uno	National Institute of Informatics (NII), Japan
Takehisa Yairi	University of Tokyo, Japan
Tamir Tassa	The Open University, Israel
Taneli Mielikainen	Nokia Research Center, USA
Tao Chen	Shenzhen Institutes of Advanced Technology, Chinese Academy of Science, China
Tao Li	Florida International University, USA
Tao Mei	Microsoft Research Asia
Tao Yang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Science, China
Tetsuya Yoshida	Hokkaido University, Japan
Thepchai Supnithi	National Electronics and Computer Technology Center, Thailand
Thomas Seidl	RWTH Aachen University, Germany
Tie-Yan Liu	Microsoft Research Asia, China
Toshiro Minami	Kyushu Institute of Information Sciences (KIIS) and Kyushu University Library, Japan

Tru Cao	Ho Chi Minh City University of Technology, Vietnam
Tsuyoshi Murata	Tokyo Institute of Technology, Japan
Vincent Lee	Monash University, Australia
Vincent S. Tseng	National Cheng Kung University, Taiwan, ROC
Vincenzo Piuri	University of Milan, Italy
Wagner Meira Jr.	Universidade Federal de Minas Gerais, Brazil
Wai Lam	The Chinese University of Hong Kong, China
Warren Jin	Australian National University, Australia
Wei Fan	IBM T.J.Watson Research Center, USA
Weining Qian	East China Normal University, China
Wen-Chih Peng	National Chiao Tung University, Taiwan, ROC
Wenjia Wang	University of East Anglia, UK
Wilfred Ng	Hong Kong University of Science and Technology, China
Wlodek Zadrozny	IBM Research
Woong-Kee Loh	Sungkyul University, Korea
Wynne Hsu	National University of Singapore, Singapore
Xia Cui	Chinese Academy of Sciences, China
Xiangjun Dong	Shandong Institute of Light Industry, China
Xiaofang Zhou	The University of Queensland, Australia
Xiaohua Hu	Drexel University, USA
Xin Wang	Calgary University, Canada
Xindong Wu	University of Vermont, USA
Xingquan Zhu	Florida Atlantic University, USA
Xintao Wu	University of North Carolina at Charlotte, USA
Xuelong Li	University of London, UK
Xuemin Lin	University of New South Wales, Australia
Yan Zhou	University of South Alabama, USA
Yang-Sae Moon	Kangwon National University, Korea
Yao Tao	The University of Auckland, New Zealand
Yasuhiko Morimoto	Hiroshima University, Japan
Yi Chen	Arizona State University, USA
Yi-Dong Shen	Chinese Academy of Sciences, China
Yifeng Zeng	Aalborg University, Denmark
Yihua Wu	Google Inc.
Yi-Ping Phoebe Chen	Deakin University, Australia
Yiu-ming Cheung	Hong Kong Baptist University, Hong Kong, China
Yong Guan	Iowa State University, USA
Yonghong Peng	University of Bradford, UK
Yu Jian	Beijing Jiaotong University, China
Yuan Yuan	Aston University, UK
Yun Xiong	Fudan University, China
Yunming Ye	Harbin Institute of Technology, China

Zheng Chen	Microsoft Research Asia, China
Zhi-Hua Zhou	Nanjing University, China
Zhongfei (Mark) Zhang	SUNY Binghamton, USA
Zhongzhi Shi	Chinese Academy of Sciences, China
Zili Zhang	Deakin University, Australia

External Reviewers

Ameeta Agrawal	York University, Canada
Arnaud Soulet	Université Francois Rabelais Tours, France
Axel Poigne	Fraunhofer IAIS, Germany
Ben Tan	Fudan University, China
Bian Wei	University of Technology, Sydney, Australia
Bibudh Lahiri	Iowa State University
Bin Yang	Aalborg University, Denmark
Bin Zhao	East China Normal University, China
Bing Bai	Google Inc.
Bojian Xu	Iowa State University, USA
Can Wang	University of Technology, Sydney, Australia
Carlos Ferreira	University of Porto, Portugal
Chao Li	Shenzhen Institutes of Advanced Technology, CAS, China
Cheqing Jin	East China Normal University, China
Christian Beecks	RWTH Aachen University, Germany
Chun-Wei Seah	Nanyang Technological University, Singapore
De-Chuan Zhan	Nanjing University, China
Elnaz Delpisheh	York University, Canada
Erez Shmueli	The Open University, Israel
Fausto Fleites	Florida International University, USA
Fei Xie	University of Vermont, USA
Gaoping Zhu	University of New South Wales, Australia
Gongqing Wu	University of Vermont, USA
Hardy Kremer	RWTH Aachen University, Germany
Hideyuki Kawashima	Nanzan University, Japan
Hsin-Yu Ha	Florida International University, USA
Ji Zhou	Fudan University, China
Jianbo Yang	Nanyang Technological University, Singapore
Jinfei	Shenzhen Institutes of Advanced Technology, CAS, China
Jinfeng Zhuang	Microsoft Research Asia, China
Jinjiu Li	University of Technology, Sydney
Jun Wang	Southwest University, China
Jun Zhang	Charles Sturt University, Australia
Ke Zhu	University of New South Wales, Australia
Keli Xiao	Rutgers University, USA
Ken-ichi Fukui	Osaka University, Japan

Kui Yu	University of Vermont, USA
Leonard K.M. Poon	Shenzhen Institutes of Advaced Technology, Chinese Academy of Science, China
Leting Wu	University of North Carolina at Charlotte, USA
Liang Du	Chinese Academy of Sciences, China
Lin Zhu	Shanghai Jiaotong University, China
Ling Chen	University of Technology Sydney, Australia
Linhao Xu	Aalborg University, Denmark
Mangesh Gupte	Google Inc.
Mao Qi	Nanyang Technological University, Singapore
Marc Plantevit	Université Lyon 1, France
Marcia Oliveira	University Porto, Portugal
Ming Li	Nanjing University, China
Mingkui Tan	Nanyang Technological University, Singapore
Natalja Friesen	Fraunhofer IAIS, Germany
Nguyen Le Minh	Japan Advanced Institute of Science and Technology, Japan
Ning Zhang	Microsoft Research Asia, China
Nuno A. Fonseca	LIACC/FEUP University of Porto, Portugal
Omar Odibat	IIIT, Hyderabad, India
Peipei Li	University of Vermont, USA
Peng Cai	East China Normal University, China
Penjie Ye	University of New South Wales, Australia
Peter Tischer	Monash University, Australia
Petr Kosina	University of Porto, Portugal
Philipp Kranen	RWTH Aachen University, Germany
Qiao-Liang Xiang	Nanyang Technological University, Singapore
Rajul Anand	IIIT, Hyderabad, India
Roberto Legaspi	Osaka University, Japan
Romain Vuillemot	INSA Lyon, France
Sergej Fries	RWTH Aachen University, Germany
Smriti Bhagat	Google Inc.
Stephane Lallich	Telecom Bretagne, France
Supaporn Spanurattana	Tokyo Institute of Technology, Japan
Vitor Santos Costa	LIACC/FEUP University of Porto, Portugal
Wang Xinchao Ecole	Polytechnique Federale de Lausanne (EPFL), Switzerland
Weifeng Su	Shenzhen Institutes of Advaced Technology, Chinese Academy of Science, China
Weiren Yu	University of New South Wales, Australia
Wenjun Zhou	Rutgers University, USA
Xiang Zhao	University of New South Wales, Australia
Xiaodan Wang	Fudan University, China
Xiaowei Ying	University of North Carolina at Charlotte, USA
Xin Liu	Tokyo Institute of Technology, Japan

Xuan Li	Chinese Academy of Sciences, China
Xu-Ying Liu	Nanjing University, China
Yannick Le	Bras Telecom Bretagne, France
Yasayuki Okabe	National Institute of Informatics, Japan
Yasufumi Takama	National Institute of Informatics, Japan
Yi Guo	Charles Sturt University, Australia
Yi Wang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Science, China
Yi Xu	SUNY Binghamton, USA
Yiling Zeng	University of Technology, Sydney
Yimin Yang	Florida International University, USA
Yoan Renaud	INSA Lyon, France
Yong Deng	Southwest University, China
Yong Ge	Rutgers University, USA
Yuan YUAN	Aston University, UK
Zhao Zhang	East China Normal University, China
Zhenjie Zhang	Aalborg University, Denmark
Zhenyu Lu	University of Vermont, USA
Zhigang Zheng	University of Technology, Sydney, Australia
Zhitao Shen	University of New South Wales, Australia
Zhiyong Cheng	Singapore Management University
Zhiyuan Chen	The Open University, Israel
Zhongmou Li	Rutgers University, USA
Zhongqiu Zhao	University of Vermont, USA
Zhou Tianyi	University of Technology, Sydney, Australia

Table of Contents – Part I

Feature Extraction

Machine Learning

Clustering

Classification

Pattern Mining

Prediction

Text Mining

Table of Contents – Part II

Graph Mining

Social Network/Online Analysis

Time Series Analysis

Sequence Analysis

Outlier Detection

Imbalanced Data Analysis

Agent Mining

Evaluation (Similarity, Ranking, Query)

Applications

An Instance Selection Algorithm Based on Reverse Nearest Neighbor

Bi-Ru Dai and Shu-Ming Hsu

The Department of Computer Science and Information Engineering,
National Taiwan University of Science and Technology, Taipei, Taiwan, ROC
brdai@csie.ntust.edu.tw, M9815067@mail.ntust.edu.tw

Abstract. Data reduction is to extract a subset from a dataset. The advantages of data reduction are decreasing the requirement of storage and increasing the efficiency of classification. Using the subset as training data is possible to maintain classification accuracy; sometimes, it can be further improved because of eliminating noises. The key is how to choose representative samples while ignoring noises at the same time. Many instance selection algorithms are based on nearest neighbor decision rule (NN). Some of these algorithms select samples based on two strategies, incremental and decremental. The first type of algorithms select some instances as samples and iteratively add instances which do not have the same class label with their nearest sample to the sample set. The second type of algorithms remove instances which do not have the same class label with their majority of kNN. However, we propose an algorithm based on Reverse Nearest Neighbor (RNN), called the Reverse Nearest Neighbor Reduction (RNNR). RNNR selects samples which can represent other instances in the same class. In addition, RNNR does not need to iteratively scan a dataset which takes much processing time. Experimental results show that RNNR achieves comparable accuracy and selects fewer samples than comparators.

Keywords: data reduction, classification, instance selection, nearest neighbor, reverse nearest neighbor.

1 Introduction

Classification [16] is one of the most popular data mining techniques [17,18], belonging to supervised learning. Supervised learning means each instance in training data with a class label. The task of classification can be divided into two parts, training and testing. Training is using training data to train a certain classifier, e.g. knearest neighbor (kNN) [19,20], support vector machine (SVM) [21] and decision tree [22], etc. Testing is using a classifier which is trained to assign a class label to a new unlabelled instance. The objective of classification is to predict the classes of unlabelled data so that users can learn which class these data belong to. The Classification results usually can support decision making and other analyses in various applications, such as disease [23], text categorization [24], search engine [25], web page [26] and image [27], etc.

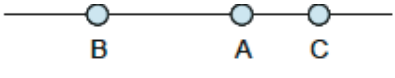
J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 1–12, 2011.

However, in nowadays, abundant data grow rapidly. If we do not process these data before classification, the storage and efficiency will become a burden. Therefore, the appearance of data reduction is for this reason. Data reduction [28,29] is to extract a subset from a dataset, so the amount of data will be reduced effectively. A way to measure the level of reduction is reduction rate. The reduction rate means the percentage of instances in training data which are chosen as samples. In spite of losing some information during the reduction process, using the subset as training data is possible to maintain classification accuracy. Instance selection [30,31] is one of data reduction approaches. The task of instance selection is to choose representative samples into the subset. However, most datasets contain noises, so we should ignore them during the instance selection process. If noises can be removed, the accuracy of using the subset for classification is possible to be higher than using the entire dataset.

Most instance selection algorithms have their own drawbacks, such as easy to preserve noises, iteratively scan a dataset and need much processing time. Therefore, the objective of this paper is to design an algorithm which can choose representative samples, ignore noises as possible and achieve acceptable processing time.

Nearest neighbor (NN) rule [1] is a lazy learning classifier in classification because it does nothing when training. However, when testing, NN needs much computation time for computing all the distances between training data and an unlabelled instance X. Then, NN finds out the nearest neighbor of X in training data and assigns its label to X. The concept of Reverse Nearest Neighbor (RNN) [2] is as follows: if A is the nearest neighbor of B and C, B and C will be in the RNN set of A. Therefore, the difference between NN and RNN is the target to be recorded. In addition, for NN, each instance has only one nearest neighbor; for RNN, an instance is possible to have zero or more than one reverse nearest neighbor in its RNN set. As shown in the example in Figure 1, A is the nearest neighbor of B and C, and C is the nearest neighbor of A. Therefore, the RNN set of A contains B and C, and the RNN set of C contains A, but there is no instance in the RNN set of B.

In this paper, we propose a instance selection algorithm, called the Reverse Nearest Neighbor Reduction (RNNR). RNNR utilizes the property of RNN to choose representative samples. The conception of RNNR is simple to understand



Instances	NN	RNN set
A	C	B,C
B	A	-
C	A	A

Fig. 1. An example of RNN

and easy to implement. RNNR applies RNN to each class and selects samples which can represent other instances in the same class. In our experiments, RNNR achieves comparable accuracy and lower reduction rate than comparators. In other words, we can use a smaller subset of the training data to obtain good classification results.

The remainder of this paper is organized as follows. In Section 2, previous approaches of instance selection are described. Section 3 presents the design of RNNR. In Section 4, experiments will show the performance of RNNR and comparators. Section 5 shows our conclusions and future work.

2 Relate Works

Most data reduction methods can be divided into two types, Instance Based Learning (IBL) algorithms [32] and Clustering Based Learning (CBL) algorithms [33]. We will focus on the former. The task of instance selection is to choose representative samples from a dataset. Many instance selection algorithms are proposed, based on NN. Some of these algorithms select samples based on two strategies, incremental and decremental. The first type of algorithms, such as Condensed NN (CNN) [6], Modified CNN (MCNN) [7], Fast NN Condensation (FCNN) [8] and Generalized CNN (GCNN) [9], select some instances as samples and iteratively add instances which do not have the same class label with their nearest sample to the sample set. The second type of algorithms, such as Edited NN (ENN) [10], remove instances which do not have the same class label with their majority of kNN. These instances will be considered as noises. Several algorithms apply ENN to their pre-processing step, such as Iterative Case Filtering (ICF) [3] and Decremental Reduction Optimization Procedure 3 (DROP3) [4]. In addition, a case-based editing algorithm, called the RDCL case profiling technique [12] and a new graph-based representation of a training set, called Hit Miss Network (HMN) [5] are also proposed to reduce the amount of data. We briefly summarize these methods as follows.

2.1 Incremental Algorithms

This type of algorithms have some characters, such as easy to preserve noises, iteratively scan a dataset, sensitive to the order of instances and generally achieve lower reduction rate than decremental algorithms.

CNN [6] randomly selects an instance as a sample and scans all instances of the training set. If any instance does not have the same class label with its nearest sample, it will be chosen as a sample. The process will continue until the remaining instances of the training set are correctly classified by samples. On the other word, each the remaining instance is absorbed by one of samples. However, CNN is possible to preserve noises and sensitive to the order of instances in a training set.

MCNN [7] solves the latter problem of CNN to generate a consistent subset. The choosing strategy of MCNN is to select one instance for each class which is

the nearest point to the geometrical center of its class as samples. This step is called Centroids. If any instance is incorrectly classified by samples, MCNN will apply the Centroids step to these misclassified instances. MCNN tends to select samples which are not close to the decision boundary.

FCNN [8] also adopts the Centroids step but makes some changes. After selecting the central instance for each class as samples, FCNN iteratively selects the nearest neighbor with different class label for each sample as samples. However, FCNN is still possible to preserve noises.

GCNN [9] like CNN adopts random selection to its initial step, but the difference is GCNN randomly selects one instance for each class as samples. Then, GCNN sets a threshold for enhancing the absorption criterion of CNN. Although GCNN achieves higher classification accuracy than CNN, GCNN selects more samples than CNN.

2.2 Decremental Algorithms

This type of algorithms have some characters, such as eliminating noises, needs more processing time and generally achieve higher classification accuracy than incremental algorithms.

ENN [10] removes instances which do not have the same class label with their majority of kNN. The objective of ENN is to remove noises in a training set. Therefore, using the sample set of ENN to classify probably achieves higher accuracy than using the entire training set. However, since ENN removes only noises, the reduction rate of ENN is hardly lower than other IBL algorithms.

After applying ENN to pre-processing step, DROP3 [4] removes the instance which is farthest from the nearest neighbor with different class label in turn. But, the hypothesis is not affecting the classification of training. Therefore, DROP3 tends to preserve instances which are close to the decision boundary.

ICF [3] applies ENN iteratively until it is impossible to remove any instance. Then, ICF removes each instance whose reachability set is smaller than coverage one. Assuming an instance X, the reachability set of X contains instances which contribute to the correct classification of X. Besides, the coverage set of X contains instances which X contributes to the correct classification of them. The above two sets is proposed by Smyth & Keane [11].

The RDCL case profiling technique [12] categorizes each case by four characteristics, R, D, C and L. R means the reachability set. D means the dissimilarity set which is proposed by Sarah [12]. C means the coverage set. L means the liability set which is proposed by Delany & Cunningham [13]. Assuming an instance X, the dissimilarity set of X contains instances which contribute to the incorrect classification of X. Besides, the liability set of X contains instances which X contributes to the incorrect classification of them. In fact, the dissimilarity set complements the reachability set, and the liability set complements the coverage set. If a certain set of X is not empty, X will possess that characteristic. Therefore, a case could possess more than one characteristic. The best removal strategy of RDCL is removing DL and DCL cases, because these cases harm to the classification of training.

HMN [5] is a new graph-based representation of a training set. Each instance of the training set has one outgoing edge for each class. Each outgoing edge points an instance to its NN with that class. The in-degree of an instance is divided into two part, hit-degree and miss-degree. Hit-degree means the edge connects two instances with the same class labels. Miss-degree means the edge connects two instances with different class labels. HMN-E removes each instance whose number of miss-degree is equal or greater than number of hit-degree. HMN-EI iteratively applies HMN-E until the classification of training is affected by removing. A drawback of HMN-EI is that it needs much processing time to iteratively construct a network for the reduced set.

3 The RNNR Algorithm

In this Section, we will introduce our instance selection algorithm, Reverse Nearest Neighbor Reduction (RNNR). The framework of RNNR is shown in Figure 2. At the beginning, training data are taken as the input of RNNR algorithm. Then, RNNR extracts a sample set from the training data. The generated sample set will be used to train a classifier. Finally, the classifier classifies some unlabelled data (testing data), based on the sample set. The final result shows the classification accuracy.

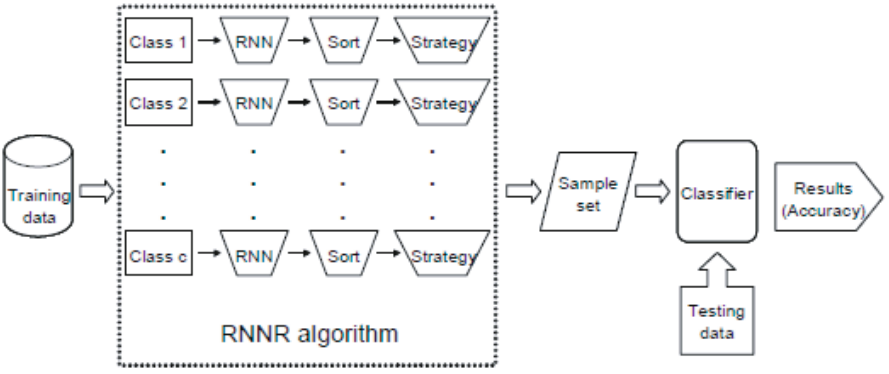


Fig. 2. The framework of RNNR

Since instances generally have similar features with their neighbors, we can select the most representative instance in its neighborhood as a sample. Furthermore, the RNN set of an instance contains instances which consider it as the nearest neighbor. Therefore, RNNR implements the above idea by applying the property of RNN. The process of RNNR algorithm is as follows. At first, RNNR applies RNN to each class. Because an instance with larger size of RNN set means it can represent more instances, RNNR will select it as a sample first. Therefore, RNNR sorts instances for each class in descending order by the size

```
RNNR algorithm
Input: A training set T
Output: A sample set S extracted from T
Initialize: Each instance of T with a RNNset = Ø and a
            Boolean value candidate = true
            S = Ø;
for each (class Ci ( i= 1, 2, ..., c) of T)
    RNN (Ci);
    Sort (Ci); // Sorting in descending order by the size of RNN set
    for each (X ε Ci)
        if X.RNNset.size > 0 and X.candidate == true /*RNNR-AL0*/
        if X.RNNset.size > 1 and X.candidate == true /*RNNR-AL1*/
        if X.RNNset.size > 1 /*RNNR-L1*/
            S = S ∪ X;
            for each (Y ε X.RNNset)
                Y.candidate = false;
Return (S);
```

Fig. 3. The pseudo code of RNNR

of RNN set. According to different choosing strategy, RNNR has three versions, RNNR-AL0, RNNR-AL1 and RNNRL1. Finally, RNNR gathers samples from each class and generates a final sample set. The pseudo code of RNNR algorithm is shown in Figure 3. Then, we will introduce choosing strategies for three versions of RNNR.

3.1 The Choosing Strategy of RNNR-AL0 (Absorption, Larger Than ZERO)

We observed that each instance whose size of RNN set is zero can not represent any instance, so RNNR-AL0 will not select it as a sample. Furthermore, noises tend to be closer to instances with different class labels. It is possible that no instance considers the noise as the nearest neighbor in the same class. Therefore, the size of RNN set of a noise is probably zero. If an instance belongs to the RNN set of a sample, it also will not be selected as a sample, owing to being absorbed by the sample. In fact, the upper bound of reduction rate of RNNR-AL0 is 50% because each sample can absorb at least one instance.

Example. The choosing strategy of RNNR-AL0 is illustrated by the training set in Figure 4. For class 1, after selecting A as a sample, B, C and D will not be selected as samples because they are in the RNN set of A. Therefore, only one sample, A, is selected for class 1. For class 2, after selecting E as a sample, F and G will not be selected as samples because they are in the RNN set of E. Next, the size of RNN set of H is larger than zero, so H will be selected as a sample. Because I is in the RNN set of H, I will not be selected as a sample. Therefore, there are two samples, E and H, in class 2. For class 3, after selecting J as a sample, because K and L are in the RNN set of J, they will not be selected as samples. Then, the size of RNN set of M is not larger than zero, so M will not be selected as a sample, too. Therefore, there is only one sample, J, in class 3. The final sample set of RNNR-AL0 is shown in Table 1.

3.2 The Choosing Strategy of RNNR-AL1 (Absorption, Larger Than ONE)

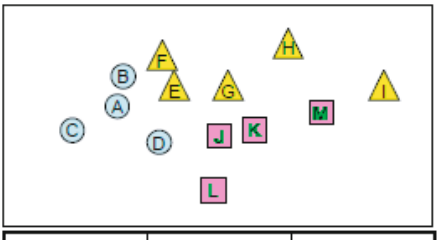
Note that each instance whose size of RNN set is one is considered as the nearest neighbor by only one instance. For selecting samples which are more representative, RNNR-AL1 will further not select each instance whose size of RNN set is one as a sample. Similar to RNNR-AL0, if an instance belongs to the RNN set of a sample, it also will not be selected as a sample, owing to being absorbed by the sample.

Example. The choosing strategy of RNNR-AL1 is illustrated by the training set in Figure 4. For class 1, after selecting A as a sample, B, C and D will not be selected as samples because they are in the RNN set of A. Therefore, only one sample, A, is selected for class 1. For class 2, after selecting E as a sample, F and G will not be selected as samples because they are in the RNN set of E. Next, H and I do not contain more than one member in their RNN sets, so they will not be selected as samples, too. Therefore, there is only one sample, E, in class 2. For class 3, after selecting J as a sample, K and L will not be selected as samples because they are in the RNN set of J. Then, the size of RNN set of M is not larger than one, so M will not be selected as a sample, too. Therefore, there is only one sample, J, in class 3. The final sample set of RNNR-AL1 is shown in Table 1.

3.3 The Choosing Strategy of RNNR-L1 (Selecting All, Larger Than ONE)

For observing how the absorption strategy affects the classification accuracy, RNNRL1 does not adopt the absorption strategy but selects each instance whose size of RNN set is larger than one as a sample. In general case, most instances contain zero or one member in their RNN sets, so the reduction rate of RNNR-L1 is still decent.

Example. The choosing strategy of RNNR-L1 is illustrated by the training set in Figure 4. Since A, E, J and K contain more than one member in their RNN sets, they will be selected as samples. The final sample set of RNNR-L1 is shown in Table 1.



Instances (Class 1)	NN	RNN set
A	B	B,C,D
B	A	A
C	A	-
D	A	-

Instances (Class 2)	NN	RNN set
E	F	F,G
F	E	E
G	E	H
H	G	I
I	H	-

Instances (Class 3)	NN	RNN set
J	K	K,L
K	J	J,M
L	J	-
M	K	-

Fig. 4. A training set for illustrating the choosing strategies of RNNR. These different shapes in the figure indicate instances in different classes. Instances are sorted in descending order by the size of RNN set for each class.

Note that all samples selected by RNNR-AL1 can be obtained from the sample set of RNNR-AL0 via removing each sample whose size of RNN set is one. Furthermore, because RNNR-L1 selects each instance whose size of RNN set is larger than one as a sample, all samples selected by RNNR-AL1 are also selected by RNNR-L1. Therefore, RNNR-AL1 should achieve the lowest reduction rate among three versions of RNNR. A problem for RNNR-AL1 and RNNR-L1 is that they will not select any instance as a sample if the RNN set of each instance contains just one member. In fact, the opportunity of the above condition is quite low.

Table 1. Sample sets which are extracted from the training set in Figure 4

Algorithm	Sample set
RNNR-AL0	A,E,H,J
RNNR-AL1	A,E,J
RNNR-L1	A,E,J,K

4 Experiments

To compare our algorithm RNNR with comparators, ICF [3], DROP3 [4] and HMN-EI [5], we use seven real datasets from UCI repository [14]. These datasets are summarized in Table 2. For each dataset, we perform five-fold cross validation to evaluate the performance. The implementation of comparators is obtained from Elena [5]. After sampling from training data for each algorithm, we use the IB1 algorithm (NN) of Weka [15] to test accuracy. The average accuracy and the reduction rate for each dataset are shown in Table 3 and Table 4. The reduction rate means the percentage of instances in training data which are chosen as samples, as shown in following Equation.

$$Reduction\ rate = No.\ of\ Samples\ /\ No.\ of\ Training\ data$$

Table 2. Datasets used in the experiments from UCI repository

Dataset	No. of Instances	No. of Attributes	No. of Classes
Wine	178	13	3
Iris	150	4	3
Bupa	345	6	2
Pima	768	8	2
Waveform	5000	21	3
Spambase	4601	57	2
Breast-W	683	9	2

The experimental results, including the classification accuracy and the reduction rate, are shown in Table 3 and Table 4. RNNR-AL1 achieves the highest accuracy among all comparators on three datasets, as shown in Table 3. In addition, three versions of RNNR have better performance in accuracy than ICF and DROP3 (comparable with HMN-EI). Especially, compared with training data (TRAIN), RNNR-AL1 and RNNR-L1 improve the accuracy on four datasets. Although RNNRAL1 selects fewer samples than RNNR-AL0, experimental results show that RNNRAL1 generally achieves higher accuracy than RNNR-AL0. This confirms that samples selected by RNNR-AL1 are more representative. In addition, RNNR-L1 achieves only slightly higher accuracy than RNNR-AL1 on most datasets, so adopting the absorption strategy can maintain the classification accuracy while achieving lower reduction rate.

For reduction rate, RNNR-AL1 selects the fewest samples of all comparators on five datasets, as shown in Table 4. In Section 3, we infer that RNNR-AL1 should achieve the lowest reduction rate among three versions of RNNR, because all samples selected by RNNR-AL1 will also be selected by RNNR-AL0 and RNNR-L1. The experimental results confirm that RNNR-AL1 indeed achieves lower reduction rate than RNNR-AL0 and RNNR-L1. Furthermore, the reduction rate of each version of RNNR is consistent on all datasets. On the contrary,

Table 3. Results of experiments (average accuracy). For each dataset, the bold value represents the highest accuracy achieved by using the data reduction algorithm. These bold values in column TRAIN mean the classification accuracy of using training data is higher than all reduction algorithms.

Dataset	RNNR-AL0	RNNR-AL1	RNNR-L1	ICF	DROP3	HMN-EI	TRAIN
Wine	93.29	**96.62**	**96.62**	93.86	93.81	94.94	96.08
Iris	95.33	**96**	**96**	91.33	94.66	**96**	95.33
Bupa	58.26	**60.29**	58.55	59.42	55.65	55.65	**60.58**
Pima	68.1	68.49	69.14	67.45	68.89	**72.14**	69.53
Waveform	77	77.38	77.42	73.02	75.54	**81.82**	76.9
Spambase	87.39	85.66	86.65	85.39	86.92	**87.52**	**90.44**
Breast-W	**97.21**	96.48	96.63	95.01	95.02	96.48	95.75

Table 4. Results of experiments (reduction rate). For each dataset, the bold value represents the lowest reduction rate achieved by using the data reduction algorithm.

Dataset	RNNR-AL0	RNNR-AL1	RNNR-L1	ICF	DROP3	HMN-EI
Wine	33.94	**20.28**	25.07	28.03	30.85	49.72
Iris	30.83	**21.67**	28	39.5	32.17	24.5
Bupa	30.29	**18.84**	26.59	24.86	25.58	23.11
Pima	32.09	**19.38**	25.86	20.46	26.74	31.82
Waveform	23.59	15.04	24.61	**12.52**	24.15	24.24
Spambase	30.07	**19.1**	24.8	25.52	30.67	35.19
Breast-W	22.05	14.65	18.65	**5.42**	25.97	35.46

the reduction rate of ICF and HMN-EI are easily affected by characteristics of a dataset.

In summary, RNNR-AL1 and RNNR-L1 generally achieve higher accuracy than ICF and DROP3 (comparable with HMN-EI). Moreover, the reduction rate of RNNRAL1 is the lowest on most of datasets. Therefore, applying RNNR-AL1 to data reduction can more effectively decrease the requirement of storage and increase the efficiency of classification, while maintaining the classification accuracy.

5 Conclusions and Future Work

In this paper, we proposed a new instance selection algorithm based on Reverse Nearest Neighbor (RNN), called Reverse Nearest Neighbor Reduction (RNNR). According to different choosing strategies, we designed three versions for RNNR.

Experiments indicated RNNR-AL1 and RNNR-L1 achieved comparable accuracy with HME-EI and better than ICF and DROP3. On more than half of datasets, RNNRAL1 and RNNR-L1 even improved the accuracy of NN. For reduction rate, RNNRAL1 had the best performance among all comparators. Therefore, applying RNNRAL1 to data reduction can more effectively decrease the requirement of storage and increase the efficiency of classification, while maintaining the classification accuracy.

Afterwards, we will extend RNN set to RkNN set, that is, find reverse k-nearest neighbors for each instance in the same class. Another idea is to apply RNN to all instances in different classes. Finally, we will observe whether the above ideas improve RNNR.

References

1. Cover, T., Hart, P.: Nearest Neighbor Pattern Classification. IEEE Trans. Information Theory 13, 21–27 (1967)
2. Korn, F., Muthukrishnan, S.: Influence sets based on reverse nearest neighbor queries. In: 19th ACM SIGMOD International Conference on Management of Data, pp. 201–212 (2000)
3. Brighton, H., Mellish, C.: On the consistency of information filters for lazy learning algorithms. In: Żytkow, J.M., Rauch, J. (eds.) PKDD 1999. LNCS (LNAI), vol. 1704, pp. 283–288. Springer, Heidelberg (1999)
4. Wilson, D.R., Martinez, T.R.: Instance pruning techniques. In: 14th International Conference on Machine Learning, pp. 403–411 (1997)
5. Marchiori, E.: Hit Miss Networks with applications to instance selection. The Journal of Machine Learning Research 9, 997–1017 (2008)
6. Hart, P.: The Condensed Nearest Neighbor Rule. IEEE Trans. Information Theory 14, 515–516 (1968)
7. Devi, F.S., Murty, M.N.: An Incremental Prototype Set Building Technique. Pattern Recognition 35(2), 505–513 (2002)
8. Angiulli, F.: Fast nearest neighbor condensation for large data sets classification. IEEE Transactions on Knowledge and Data Engineering 19, 1450–1464 (2007)
9. Chou, C.H., Kuo, B.H., Chang, F.: The generalized condensed nearest neighbor rule as a data reduction method. In: 18th International Conference on Pattern Recognition, pp. 556–559 (2006)
10. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man and Cybernetics SMC-2, 408–421 (1972)
11. Smyth, B., Keane, M.: Remembering to forget: A competence preserving case deletion policy for CBR systems. In: Mellish, C. (ed.) 14th International Joint Conference on Artificial Intelligence, pp. 337–382. Morgan Kaufmann, San Francisco (1995)
12. Delany, S.J.: The Good, the Bad and the Incorrectly Classified: Profiling Cases for Case-Base Editing. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 135–149. Springer, Heidelberg (2009)
13. Delany, S.J., Cunningham, P.: An analysis of case-base editing in a spam filtering system. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 128–141. Springer, Heidelberg (2004)
14. University of California, Irvine. Machine Learning Repository, http://archive.ics.uci.edu/ml/

15. University of Waikato. Weka Machine Learning Project, http://www.cs.waikato.ac.nz/ml/weka/
16. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley Interscience, Hoboken (2000)
17. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
18. Tan, P., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley, Reading (2005)
19. Keller, J.M., Gray, M.R., Givens, J.A.J.R.: A fuzzy K-nearest neighbor algorithm. IEEE Transactions on Systems, Man, and Cybernetics 15(4), 580–585 (1985)
20. Seidl, T., Kriegel, H.P.: Optimal multi-step k-nearest neighbor search. ACM SIGMOD Record 27(2), 154–165 (1998)
21. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2(2), 121–168 (1998)
22. Quinlan, J.R.: Induction of decision trees. Machine Learning 1(1), 81–106 (1986)
23. Knaus, W.A., Draper, E.A., Wagner, D.P., Zimmerman, J.E.: APACHE II: a severity of disease classification system. Crit Care Med. 13(10), 818–829 (1985)
24. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In: 16th International Conference on Machine Learning, pp. 200–209 (1999)
25. Joachims, T.: Optimizing search engines using click through data. In: 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, SESSION: Web Search and Navigation, pp. 133–142 (2002)
26. Dumais, S., Chen, H.: Hierarchical classification of Web content. In: 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 256–263 (2000)
27. Haralick, R.M., Shanmugam, K., Dinstein, Its'Hak: Textural Features for Image Classification. IEEE Transactions on Systems, Man and Cybernetics 3(6), 610–621 (1973)
28. Bevington, P.R., Robinson, D.K.: Data Reduction and Error Analysis for the Physical Sciences, 3 edn. (2002)
29. Tonry, J., Davis, M.: A survey of galaxy redshifts. I - Data reduction techniques. Astronomical Journal 84, 1511–1525 (1979)
30. Cano, J.R., Herrera, F., Lozano, M.: Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. IEEE Transactions on Evolutionary Computation 7(6), 561–575 (2003)
31. Brighton, H., Mellish, C.: Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery 6(2), 153–172 (2002)
32. Aha, D., Kibler, D., Albert, M.K.: Instance-Based Learning Algorithms. Machine Learning 6(2), 37–66 (1991)
33. Dunn, J.C.: Some recent investigations of a new fuzzy partition algorithm and its application to pattern classification problems. J. Cybernetics 4, 1–15 (1974)

A Game Theoretic Approach for Feature Clustering and Its Application to Feature Selection

Dinesh Garg[1], Sellamanickam Sundararajan[1], and Shirish Shevade[2]

[1] Yahoo! Labs, Bangalore, India
{dineshg,ssrajan}@yahoo-inc.com
[2] Department of CSA, IISc, Bangalore, India
shirish@csa.iisc.ernet.in

Abstract. In this paper, we develop a game theoretic approach for clustering features in a learning problem. Feature clustering can serve as an important preprocessing step in many problems such as feature selection, dimensionality reduction, etc. In this approach, we view features as rational players of a coalitional game where they form coalitions (or clusters) among themselves in order to maximize their individual payoffs. We show how Nash Stable Partition (NSP), a well known concept in the coalitional game theory, provides a natural way of clustering features. Through this approach, one can obtain some desirable properties of the clusters by choosing appropriate payoff functions. For a small number of features, the NSP based clustering can be found by solving an integer linear program (ILP). However, for large number of features, the ILP based approach does not scale well and hence we propose a hierarchical approach. Interestingly, a key result that we prove on the equivalence between a k-size NSP of a coalitional game and minimum k-cut of an appropriately constructed graph comes in handy for large scale problems. In this paper, we use feature selection problem (in a classification setting) as a running example to illustrate our approach. We conduct experiments to illustrate the efficacy of our approach.

1 Introduction

In many supervised and unsupervised learning problems, one often needs to conduct a preprocessing step on a feature set representing the input to make the learning task feasible. Having some insights about usefulness of the features can add value to finding better solutions. For example, imagine a classification problem where the number of features is so large that it is not possible to train any satisfactory model in allotted time with available resources. In such a situation, one needs to select a subset of features on which a model can be trained in a reasonable amount of time without significant degradation in generalization performance. This problem is known as the *feature selection* problem [12], [9], [17]. We believe that if we can group the features according to the following two criteria then it will not only make the task of feature selection easy but also

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 13–25, 2011.

would greatly improve the quality of the features selected and hence the final generalization performance.

(1) Relevant Features vs Irrelevant Features: For a classification task, a feature f_i is a relevant feature if removal of f_i alone will result in performance deterioration of an optimal classifier. Otherwise, we call it an irrelevant feature. See Kohavi and John [12] for a detailed discussion.

(2) Substitutable vs Complementary Features: For a classification task, we characterize two features f_i and f_j as substitutable features if there is no significant difference in the generalization performance of a classifier that is trained using both the features and the generalization performance of a classifier that is trained using just (any) one of these two features. On the other hand, if the generalization performance of the first classifier is significantly better than that of the latter one then we attribute these two features as complementary features.

It is easy to see that having the above insights about the features would greatly simplify the task of feature selection - *one just needs to select a set of features of the desired size in such a manner that all the features in that set are* relevant *and* complementary *to each other*. This insight about the features can be obtained by using the feature clustering approach proposed in this paper.

In this paper, we develop a novel game theoretic approach for clustering the features where the features are interpreted as players who are allowed to form coalitions (or clusters)[1] among themselves in order to maximize their individual payoffs (defined later). In this approach, the choice of a payoff function would determine whether all the features within a cluster would be substitutable or complementary to each other. It is important to mention that although we demonstrate the feature clustering approach through the idea of substitutable and complementary features, the approach is quite generic and can be used in other problems like dimensionality reduction and community detection in web graphs [7]. For any other problem, one needs to select a payoff function appropriately so that the desired insights become apparent in clustering the features. We believe that this game theory based approach for feature clustering is quite novel and unique till date.

The key contributions of this paper are as follows: (1) We draw an analogy between a coalitional game and feature clustering problem and then show that Nash Stable Partition (NSP), a well known solution concept in the coalitional game theory [15], provides a natural way of clustering the features (Section 3); (2) We show how to get an NSP based clustering by solving an integer linear program (ILP) (Section 3). The solution of this ILP gives an NSP based clustering of the features for a given payoff function under the assumption of *pure hedonic* setting (defined later); (3) We illustrate that depending upon how a payoff function is chosen, the NSP based clustering would have a property that either substitutable features are grouped together in one cluster or complementary features are grouped together in one cluster (Section 4). Then, one can use any standard technique [9], [17] to

[1] We will keep using the terms coalition and cluster interchangeably.

choose features from these clusters. We also suggest a simple cluster ranking based technique for selecting the features from a given set of feature clusters (Section 5); (4) Finally, we propose a hierarchical scheme for feature clustering in the scenario where the feature set size is very large and solving ILP is expensive (Section 6). Interestingly, a key result that we prove on the equivalence between a k-size NSP of a coalitional game and minimum k-cut of an appropriately constructed graph [13],[4] comes in handy for large scale problems; (5) We demonstrate the efficacy of our approach through a set of experiments conducted on real world as well as synthetic datasets (Section 7).

2 Related Work

Feature selection can be thought of as a dimensionality reduction technique. The problem of feature selection is to find a feature subset S with m features, which jointly have the largest dependency on the target class. Feature selection methods can be of two types: *filter* methods and *wrapper* methods [12]. Filter methods select the relevant features by ranking all the features using some measure. On the other hand, the wrapper methods treat an induction algorithm as a black box and interact with it to assess the usefulness of a subset of features.

Different measures have been suggested in the literature for ranking the features. See Guyon and Elisseeff [9] and the references therein. Some commonly used measures are the absolute value of the Pearson correlation coefficient and mutual information [17]. Peng et al [17] studied the feature selection problem using maximum statistical dependency criterion based on the mutual information. Particularly relevant to our work is the approach suggested by Cohen et al [3], which poses the problem of feature selection as a cooperative game problem.

Cohen et al [3] proposed to use Shapley value (a solution concept for cooperative games) for ranking the features. In this approach, each feature is treated as a player of a game and the idea is to evaluate the marginal contribution of every feature to the classification accuracy by using Shapley value and then eliminate those features whose marginal contribution is less than a certain threshold. Since the calculation of the Shapley value involves summing over all possible permutations of the features, it becomes impractical to determine the Shapley value if the feature set size is large. It was therefore proposed to use multi perturbation approach of approximating the Shapley value (proposed by Keinan et al [11]). The main drawback of this approach is that it is computationally expensive as different classifiers need to be trained using different feature subsets.

3 Coalitional Games Preliminaries

We start with definitions of a few basic concepts in the coalitional game theory which would serve as building blocks for the rest of the paper. These concepts are fairly standard in the game theory literature and can be found in [8].

Definition 1 (Coalitional Games). *An n-person coalitional game (under pure hedonic setting) is a pair* $(N, u(\cdot))$*, where* $N = \{x_1, x_2, \ldots, x_n\}$ *is the set of players and* $u(\cdot) = (u_1(\cdot), u_2(\cdot), \ldots, u_n(\cdot))$ *is the profile of players' utility functions. Utility function* $u_i(\cdot)$ *is defined over the set of coalitions* $\{C \subset N | i \in C\}$ *and* $u_i(C)$ *signifies the utility of the player* x_i *if he decides to join the coalition* C*.*

The pure hedonic setting characterizes a special class of coalitional games where the utility of a player due to joining a coalition depends only on the composition of the coalition. The term *pure hedonic setting* was coined by [6]. The examples of hedonic setting include the formation of social clubs and political parties. In the rest of this paper, we will be working with coalitional games under the pure hedonic setting, which we refer to as hedonic games.

The key question addressed by the coalitional game theory is that for a given coalitional game, what coalitional structure would emerge if the players play the game. A coalition structure $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$ is a partitioning of the players into k disjoint sets $C_1, \ldots, C_k$. Many stability concepts have been proposed in the coalitional game theory [2]. In this paper we focus only on *Nash Stability.*

Definition 2 (Nash Stable Partition (NSP)). *Given a hedonic game* $(N, u(\cdot))$*, a partition* $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$ *is Nash stable if for every player* i*, we have* $u_i(C_{\mathscr{C}}(i)) \geq u_i(C_j \cup \{i\})\ \forall C_j \in \mathscr{C} \cup \{\varnothing\}$*, where* $C_{\mathscr{C}}(i)$ *denotes the set* $C_j \in \mathscr{C}$ *such that* $i \in C_j$*. In simple words, a partition* $\mathscr{C}$ *is an NSP if no player can benefit from switching his current coalition* $C_{\mathscr{C}}(i)$ *given that all the other players are sticking to the coalitions suggested by the partition* $\mathscr{C}$*. The NSP where* $\mathscr{C} = \{N\}$ *is trivial NSP and any other NSP is called a non-trivial NSP. A non-trivial NSP* $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$ *is called a k-size NSP (or k-NSP for short).*

The obvious question that arises next is that whether an NSP always exists and if not then under what conditions it exists. In general, it is possible that a given hedonic game may not have any NSP. However, for some classes of hedonic games, existence of an NSP is guaranteed (e.g., the games with *additively separable (AS) and symmetric preferences* (defined next)).

Definition 3 (AS and Symmetric Preferences). *A player* i*'s preferences are additively separable if there exists a function* $v_i : N \rightarrow R$ *such that* $\forall\ C \subseteq N$ *for which* $i \in C$*, we have* $u_i(C) = \sum_{j \in C} v_i(j)$*. Without loss of generality, we can set* $v_i(i) = 0$ *which would imply that* $u_i(C) = u_i(C \cup \{i\})\ \forall C \subseteq N$*. Thus, we can say that any additively separable preferences can be equivalently represented by an* $n \times n$ *matrix* $v = [v_i(j)]$*. Further, we say that the AS preferences satisfy symmetry iff the matrix* v *is symmetric, that is* $v_i(j) = v_j(i) = v_{ij}\ \forall i, j$*.*

It turns out that the problem of deciding whether an NSP exists in a pure hedonic game with the AS preferences is NP-complete [1], [16]. However, Bogomolnaia and Jackson [2] have shown that an NSP exists in every hedonic game having the AS and *symmetric* preferences. It is worth mentioning that Bogomolnaia and Jackson proved the existence of an NSP. But their result does not specify

anything regarding the structure of the NSP (e.g., whether the NSP would be trivial or non-trivial, whether there is a unique NSP or many, what is the size of such NSP(s), etc). However, Olsen [16] has shown that the problem of deciding whether a non-trivial NSP exists for the AS hedonic games with non-negative and symmetric preferences is NP-complete. In the rest of the paper, we will be working under the hedonic setting with the AS and symmetric preference assumptions and hence we do not have to worry about the existence of an NSP (although it could be a trivial NSP).

In what follows, we extend the notion of an NSP to an approximate NSP which we prefer to call as ϵ-Regret NSP. The motivation behind this notion comes from the facts that (1) a hedonic game with the AS and symmetric preferences may not have a non-trivial NSP (i.e., a k-NSP with $k > 1$) but for any value of $1 \leq k \leq n$, there always exists an $\epsilon \geq 0$ such that the game has an ϵ-Regret k-NSP, and moreover, (2) checking for the existence of a non-trivial NSP is NP-complete (under non-negative preferences) [16].

Definition 4 (ϵ-Regret k-NSP). *A given k-size partition of the players in a hedonic game is said to be ϵ-regret k-NSP if no player can increase his utility by more than ϵ by switching his current coalition given that no other player is switching his current coalition.*

3.1 NSP Computation via Integer Linear Program (ILP)

Theorem 1. *Consider an n-person hedonic game having the AS and symmetric preferences given by a matrix $v = [v_{ij}]$. Let $\mathscr{C}^*$ be a partition of the set N of players. If $\mathscr{C}^*$ is a solution of the following ILP then $\mathscr{C}^*$ is an NSP of this game.*

$$\begin{aligned} &\textit{maximize } \sum\nolimits_{C \subset N} \alpha(C) v(C) \\ &\textit{subject to } \sum\nolimits_{C \subset N | i \in C} \alpha(C) = 1 \; \forall \; i \in N; \alpha(C) \in \{0,1\} \; \forall C \subset N \end{aligned} \tag{1}$$

where $v(C) = \sum_{i,j|i,j \in C} v_{ij}$.

In the above ILP, $\alpha(C)$ is an indicator variable that decides whether the coalition C is a part of the partition or not. The constraints basically ensure a partition of the players. The objective function is nothing but the sum of values of all the coalitions in a partition, which we call as *partition's value*. Thus, we can say that any value maximizing partition is always an NSP. Note that the above theorem provides a sufficiency condition for an NSP and it can be directly obtained from the existence proof of [2]. However, it is not necessary that if a non-trivial partition $\mathscr{C}$ of N is an NSP then that would also be a solution of the above ILP. For example, consider a 4×4 symmetric matrix v having $v_{ii} = 0 \; \forall i$; $v_{12} = 10$; $v_{13} = v_{14} = 1$; $v_{23} = v_{24} = 1$; $v_{34} = 5$. It is easy to see that for this matrix we have $\mathscr{C} = \{\{1,2\},\{3,4\}\}$ as an NSP but this is not the optimal solution of the ILP. The optimal solution of the ILP is a trivial NSP. This observation can be generalized by saying that if the matrix v is nonnegative then the above ILP will give only the trivial partition as the solution. Note that solving an ILP is in

general hard and quite impractical (especially for a large number of variables). Hence, one can work with an LP relaxation ($0 \leq \alpha(C) \leq 1$) of the above ILP and get an approximate solution. In fact, in our feature clustering approach (proposed in the next section), we use an LP relaxation of this ILP.

4 Feature Clustering via Nash Stable Partition

In this section, we show that the feature clustering problem can be effectively posed as a Nash stable partitioning problem. For this, let us consider the binary classification problem $\{x_l, y_l\}_{l=1}^m$ where we have m training examples and the input vector x_l is specified in the form of n real valued features $\{f_1, f_2, \ldots, f_n\}$ and $y_l \in \{-1, +1\}$. Let ρ_{ij} be the estimate of the Pearson correlation coefficient between the feature f_i and the feature f_j. Similarly, ρ_{iy} is the Pearson correlation coefficient between the feature f_i and the class label y. Now let us set up an analogy between a feature cluster and an NSP as follows. View each feature as a player in the game and define a payoff function $v_i(j) = |\rho_{ij}|\ \forall i, j \in N$. Define $v_i(i) = 0\ \forall\ i$. It is easy to see that $v_i(j) = v_j(i) = v_{ij} = |\rho_{ij}|$. Assume that each feature f_i has a preference relation $\succeq_i$ over the feature subsets such that $\forall C_1, C_2 \ni f_i$, we have $C_1 \succeq_i C_2 \Leftrightarrow \sum_{j \in C_1} v_i(j) \geq \sum_{j \in C_2} v_i(j)$. If the features are allowed to form the coalitions then every feature would tend to join a feature group which maximizes its payoff function. It is easy to see that for the chosen function $v_{ij} = |\rho_{ij}|$, substitutable features would tend to group together. The above situation can be viewed as a coalitional game with the AS and symmetric preferences for which an NSP is a reasonable solution concept to predict the final coalitional structure. Therefore, if we use an NSP of the above game as clusters of the features then it would have the property that the features are stable in their own clusters and don't want to move across the clusters (which is a desired property of any clustering scheme). Thus, we can say that any NSP of the above game would be a reasonable clustering of the features where each cluster would contain substitutable features and the features across clusters would be complementary to each other. A Nash stable partition of the above game can be obtained by solving ILP given in (1). Recall, if v_{ij} is nonnegative for all i and j then ILP does not have a non-trivial solution and moreover computing a non-trivial NSP is NP-complete due to the result of Olsen [16]. In order to tackle this situation, we can modify the payoff functions slightly as follows: $v_{ij} = |\rho_{ij}| - \beta$ where $0 < \beta < 1$. β can be viewed as a parameter which decides a threshold (in an implicit manner) such that any two features having v_{ij} values higher (lower) than the threshold would qualify as substitutable (complementary) features. With this interpretation, it is easy to see that as we increase β, the threshold increases and hence we get more and more fine grained clusters of substitutable features.

It is interesting to note that the function $v_{ij} = |\rho_{ij}| - \beta$ is not the only payoff function but there exist many other choices and depending upon the function we decide to choose, we may either get substitutable features grouped together or complementary features grouped together. For example, if we use

$v_{ij} = |\rho_{iy}|+|\rho_{jy}|-|\rho_{ij}|-\beta$ where $0 < \beta < 2$ then complementary features would tend to group together and each cluster will contain relevant and complementary features. Note that the maximum possible value of the function $|\rho_{iy}|+|\rho_{jy}|-|\rho_{ij}|$ is 2 and having $\beta > 2$ would make all $v_{ij},\ i \neq j$ negative. Because $v_{ii} = 0\ \forall i$, this would result in a solution where each single feature would form its own cluster. This is the reason why we have restricted the value of β upto 2. One can also work with the function $v_{ij} = |\rho_{iy}| + |\rho_{jy}| + |\rho_{ij}| - \beta$ where $0 < \beta < 3$. This will again result in substitutable features getting grouped together. In each one of these functions, one can replace $|\rho_{ij}|$ with m_{ij} and $|\rho_{iy}|$ with m_{iy} where m_{ij} is the estimate of the mutual information between the features f_i and f_j. Below, we suggest a few functions along with their clustering nature.

Substitutable Features: $(|\rho_{ij}| - \beta)$, $(|\rho_{iy}| + |\rho_{jy}| + |\rho_{ij}| - \beta)$, $(m_{ij} - \beta)$, and $(m_{iy} + m_{jy} + m_{ij} - \beta)$.

Complementary Features: $(|\rho_{iy}|+|\rho_{jy}|-|\rho_{ij}|-\beta)$ and $(m_{iy}+m_{jy}-m_{ij}-\beta)$.

5 Feature Selection Approaches

Here we discuss a few approaches for selecting a given number of features. In the first approach, we choose a payoff function that puts substitutable features in one cluster. Next, we tune the parameter β in such a way that we obtain m feature clusters as a solution of ILP (or relaxed ILP). Now we pick the most relevant feature from each cluster. The most relevant feature of a cluster can be chosen by several schemes such as maximum $|\rho_{iy}|$ or maximum m_{iy}. Note that in this approach, there is a risk of getting an irrelevant feature selected if all the features in that group have very low value of $|\rho_{iy}|$. Therefore, we propose an alternative approach next.

In the second approach also, we choose a payoff function that puts the substitutable features in one cluster. Next, we choose some value of β and obtain the feature clusters as the solution of ILP (or relaxed ILP). In this approach, we have no control over the number of clusters as β is not tuned. We compute a ranking score $r(C)$ for each cluster C in the partition and then pick the top cluster. From this top cluster, we pick the most relevant feature (using the criterion discussed in the previous approach) and then delete that feature from this cluster. Now we recompute the ranking score for this cluster and perform the feature selection step as before. We repeat this whole process until we obtain the desired number of features. A few suggested ranking functions are $r(C) = (\sum_{i \in C} |\rho_{iy}|)/|C|$; $r(C) = (\sum_{i \in C} m_{iy})/|C|$. One can develop similar approaches by using a payoff function for grouping complementary features together.

6 Handling of Large Feature Set Size

In section 4, we suggested that in most of the cases, the LP relaxation of ILP gives an approximate clustering based on NSP. However, it is easy to see that

even the relaxed LP would have 2^n variables and hence running time of any LP solver is large even for as small a number of features as $n = 20$. In such a scenario, ILP or relaxed ILP based approaches are not feasible for feature clustering. To tackle such situations, we propose a hierarchical feature clustering approach. This approach is based on an interesting result on the equivalence between a k-NSP of a coalitional game and minimum k-cut of an appropriately constructed graph (as we prove below).

6.1 Equivalence between a k-NSP and Minimum k-Cut

Consider an n-person hedonic game with the AS, symmetric, and non-negative preferences given by an $n \times n$ symmetric and non-negative matrix $v \geq 0$. This game can be represented by an undirected graph $G = (N, E)$ where N is the set of nodes which is the same as the set of players. We put an undirected edge between node i and j iff $v_{ij} = v_{ji} > 0$ and assign a weight v_{ij} to that edge.

Definition 5 (Minimum k-Cut). *A k-cut of a graph G is defined as any partitioning of the nodes into k $(1 < k \leq n)$ nonempty disjoint subsets, say $\{C_1, C_2, \ldots, C_k\}$. The capacity of such a k-cut is defined as the sum of the weights of all those edges whose end points are in different partitions and is denoted by $cap(C_1, C_2, \ldots, C_k)$. A minimum k-cut of this graph is a k-cut whose capacity is minimum across all the possible k-cuts of the graph. For any given feasible value of k, there could be multiple minimum k-cuts and the capacities of all those cuts would be the same and we denote that by $cap^*(k)$.*

Definition 6 (Support Size). *The support size s of a k-cut $\{C_1, C_2, \ldots, C_k\}$ is defined as follows: $s = \min_{i=1,\ldots,k} |C_i|$.*

Theorem 2 (k-NSP and Minimum k-Cut). *Let G be a graph representation of a hedonic game with the AS and symmetric preferences given by a matrix $v \geq 0$. Then, minimum k-cut of G having $s > 1$ $\Rightarrow$ k-NSP.*

Proof: The proof is by contradiction. If possible, let $\{C_1, C_2, \ldots, C_k\}$ be a minimum k-cut with support $s > 1$ and it is not a k-NSP. This would mean that there exists some player $i \in N$ who would gain by switching to some other coalition given that no other player is switching. Let us assume that $i \in C_t$ for some $t \in \{1, 2, \ldots, k\}$. Let $u_i(C_z) > u_i(C_t)$ for some $z \neq t$ and hence player i would prefer to switch to the coalition C_z from his current coalition C_t.[2] Because of the AS and symmetric preferences, we have $u_i(C) = \sum_{j \in C} v_{ij}$ for any coalition C, and hence the cut capacity of this newly formed k-cut after the switching is given by $cap(\text{new cut}) = cap(C_1, \ldots, C_k) + u_i(C_t) - u_i(C_z)$. Because $u_i(C_z) > u_i(C_t)$, we have $cap(\text{new cut}) < cap(C_1, C_2, \ldots, C_k)$ which is a contradiction to the assumption that $\{C_1, C_2, \ldots, C_k\}$ is a minimum k-cut.[3] *(Q.E.D.)*

[2] Note that i cannot gain by isolating itself (i.e. $C_z = \emptyset$) because in that case $u_i(C_z) = v_{ii} = 0$ and $0 > u_i(C_t)$ which is a contradiction because v is non-negative.

[3] A similar result was proved by [7] in the context of web communities for $k = 2$.

In the above proof, $s > 1$ is required to ensure that even after node i switches, the resulting partition is a k-way cut. Theorem 2 gives a sufficiency condition for the existence of a k-NSP and hence becomes useful in the case when computing a minimum k-cut of a graph is an easy problem. However, it is a well known fact that computing a minimum k-cut of a graph is NP-hard for $k > 2$ [20]. Therefore, this theorem would not help much for computing a k-NSP of a given game if $k > 2$. To handle that case, it is useful to define an approximate minimum cut in the form of ϵ-Regret k-cut of a graph (defined below). In Theorem 3, we show that there exists a connection between an ϵ-Regret k-cut of a graph and an ϵ-Regret k-NSP. The proof of this theorem follows the similar line of arguments as in the proof of Theorem 2. Hence, we skip the proof.

Definition 7 (ϵ-Regret k-Cut). *Given a graph G, a k-cut $\{C_1, C_2, \ldots, C_k\}$ is said to be an ϵ-Regret k-cut iff $cap^*(k) \leq cap(C_1, C_2, \ldots, C_k) \leq cap^*(k) + \epsilon$, where $cap^*(k)$ is the capacity of the minimum k-cut.*

Theorem 3. $\boxed{\epsilon\text{-Regret } k\text{-cut having } s > 1} \Rightarrow \boxed{\epsilon\text{-Regret } k\text{-NSP}}$

6.2 Hierarchical Feature Clustering

From Theorem 3, it is apparent that any scheme that efficiently computes an ϵ-Regret k-cut of a graph would be an efficient scheme for computing the ϵ-Regret k-NSP also. Therefore, we propose a simple hierarchical scheme (without any theoretical bounds on the value of ϵ) for computing an ϵ-Regret k-cut and hence an ϵ-Regret k-NSP. Note that when v has only positive entries (and under some cases with negative entries also), it becomes a polynomial time solvable problem to compute a minimum 2-cut [19]. Hence, our approach works by computing minimum 2-cuts of a graph in a hierarchical manner.

In this approach, we begin with the whole feature set and just compute a minimum 2-cut of the underlying graph. Then we recursively compute the minimum 2-cut of each partition obtained in the previous step. This gives us a binary tree structure (not necessarily balanced) where each node is the set of features and two children of a node correspond to a minimum 2-cut of that node. We stop splitting a node further if the number of features in that node is less than some threshold value. All such nodes would form the leaf nodes. Finally, we take all the leaf nodes of such a tree as feature clusters and then apply any one of the feature selection approaches discussed earlier.

Note that at every step of the above scheme, one can use an exact algorithm for computing a minimum 2-cut (for example, $O(mn + n^2 log(n))$ algorithm by Stoer and Wagner [19] where m is the number of edges). However, it is possible to have $m = O(n^2)$ in which case finding an exact solution becomes expensive ($O(n^3)$). In addition to this, if a graph has multiple minimum 2-cuts then we would be interested in focusing on those cuts for which the support size is more than 1. Therefore, in practice one may prefer a fast randomized algorithm (although approximate) as opposed to an exact algorithm because we can run it

several times and generate various minimum 2-cut solutions and then pick the most balanced solution. Some of the methods that are fast and can generate multiple solutions include Karger and Stein's [10] randomized mincut algorithm ($O(n^2 log^3 n)$ for a cut) and spectral clustering (SC) algorithms ($O(n^3)$ for a cut). We conducted experiments with a SC algorithm for ease of implementation and to get a feel of the performance achievable using an approximate method. Although the SC algorithm has same $O(n^3)$ complexity as the Stoer and Wagner's algorithm but we found that the SC algorithm was significantly faster in our experiments. The SC algorithm which we use here is as follows [13] [5]: (1) Construct a diagonal matrix D of the size same as the matrix v in such a way that each diagonal entry of D is the sum of the entries of the corresponding row of the matrix v; (2) Compute the graph Laplacian matrix $L = D - v$; (3) Find the two eigenvectors corresponding to the two lowest eigenvalues of the matrix L; (4) Apply 2-means clustering on these two eigenvectors. These clusters would correspond to an approximate minimum 2-cut. It is important to note that in this approximate algorithm, the 2-means clustering algorithm involves a random initialization of the clusters. Depending on how the initialization is done, we may get different solutions. Among the different solutions, the final solution can be picked by using an appropriate cluster quality measure.

7 Experiments

We illustrate our approach by conducting several experiments on the feature selection problems for one synthetic dataset and two real world datasets - *Splice* and *Arrhythmia* (details given later), which are binary classification problems.

Synthetic Datasets (No. of Features = 20, Training Set Size = 1000, Test Set Size = 300): We generated synthetic datasets corresponding to 20-dimensional zero mean correlated Gaussian random variables of 15 different covariance matrices. Each of these 15 datasets consisted 1000 training samples, denoted by X_{tr}, and 300 testing samples, denoted by X_{ts}. In our experiment, we generated a coefficient vector w of size 20 for a linear classifier and used it to obtain the class labels for the training data as follows: $Y_{tr} = sign(w' \cdot X_{tr})$. In the same way, we generated the class labels for the test data. The vector w was generated as 20 i.i.d. uniform random variables over the interval $[-1, 1]$. Next we computed the correlation matrix $\rho = [\rho_{ij}]$ for the training data. We set the payoff function $v_{ij} = |\rho_{ij}| - \beta$ and then varied β from 0 to 1 so that we get different size NSPs by solving the corresponding relaxed LPs for ILP (1). For each of these NSPs, we used $|\rho_{iy}|$ to pick the relevant features and trained a linear least squares classifier on the training data using these selected features. We computed the accuracy of this classifier on the test data (with the selected features). We performed this experiment 1000 times by changing the vector w and computed the average accuracy over these realizations. The same experiment was repeated for 15 different covariance matrices. The results of our experiments are summarized in Figure 1 (top panel); we have plotted the variation of average accuracy for 15 different covariance matrices in the form of blue colored

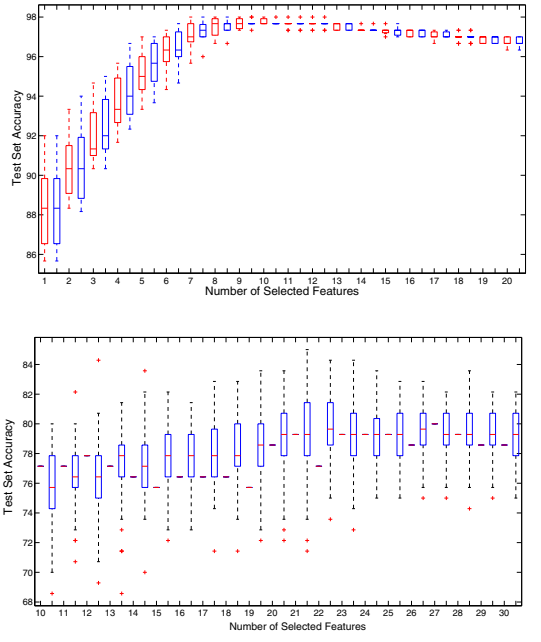


Fig. 1. Results for Synthetic and Arrhythmia Datasets

box plots (for our method). We have also plotted a red colored box plot along with each blue box plot. The red box plot corresponds to the variation of the average accuracy when the features were picked from the set of features ranked in decreasing order according to $|\rho_{iy}|$. We conducted statistical significance test using Wilcoxon sign rank test at the significance level of 0.05 to compare the two schemes of feature selection and found that for the feature set sizes of 3 to 6, the game theoretic based feature clustering approach (followed by feature selection) yielded better performance than selecting the features using $|\rho_{iy}|$ values.

Splice Datasets (No. of Features = 60, Training Set Size = 1000, Test Set Size = 2175): The splice datasets are taken from `http://theoval.cmp.uea.ac.uk/~gcc/matlab/default.html`. In this experiment our goal was to test the *relevant feature identification capability* of our method. Because the number of features is large, we used the hierarchical clustering approach (described in Section 6) for obtaining the clusters of the features and then we picked top 7 features using the second approach for feature selection (proposed in Section 5) by making use of the ranking function $r(S) = (\sum_{i \in S} |\rho_{iy}|)/|S|$. We repeated this experiment 1000 times with random initialization for 2-means clustering. We found that most of the times, 6 out of the selected 7 features belong to the following set: $\{f_{28}, f_{29}, f_{30}, f_{31}, f_{32}, f_{33}, f_{34}\}$ which is almost same as the set of relevant features identified by Meilă and Jordan [14]. This demonstrates the effectiveness of the proposed method on real world problems.

Arrhythmia Dataset (No. of Features = 246, Training Set Size =280, Test Set Size = 140): This dataset can be obtained from the UCI repository. We used a version of this dataset that was modified by Perkins et al [18] and was also used by Cohen et al [3]. For this dataset, we again performed the hierarchical feature clustering followed by the feature selection (varying the size from 10 to 30) in the same way as we did for the Splice dataset case. We repeated the whole process for 100 different initial conditions as was done in the Splice datasets case. For each case, we trained a simple linear least squares classifier on the selected features and then recomputed the accuracy of the test set. We have plotted the boxplot for the variation in accuracy in Figure 1 (bottom panel). Along with each box plot, we have indicated the test set accuracy (in the form of red dash) when the same number of features are selected from the set of features ranked in decreasing order according to $|\rho_{iy}|$ and the linear least squares classifier is trained on those features. We see that the median of the test set accuracy with the feature clustering is higher than 75% for each case (which was the reported performance of Perkins et al [18]). Specifically, for 21 features, we see that the median of the performance is 79% whereas the highest performance reaches beyond 85% (which is higher than 84.2% reported by Cohen et al [3]).

To conclude, although the proposed approach is complete in its own, there are several avenues for further investigations. For example, all the experiments were conducted using the payoff functions that put substitutable features in one cluster. One can experiment with other payoff functions which would drive complementary features in one cluster. In the case of multiple NSPs, we assumed that it is fine to choose any one of them. However, it is worth investigating the quality of other NSPs.

References

1. Ballester, C.: NP-completeness in hedonic games. Games and Economic Behavior 49(1), 1–30 (2004)
2. Bogomolnaia, A., Jackson, M.O.: The stability of hedonic coalition structures. Games and Economic Behavior 38, 201–230 (2002)
3. Cohen, S., Dror, G., Ruppin, E.: Feature selection via coalitional game theory. Neural Computation 19, 1939–1961 (2007)
4. Dhillon, I., Guan, Y., Kulis, B.: A unified view of kernel k-means, spectral clustering, and graph partitioning. Technical report, Univ. of Texas, Austin (2005)
5. Ding, C.: A tutorial on spectral clustering, `http://crd.lbl.gov/~cding/Spectral/`
6. Drèze, J., Greenberg, J.: Hedonic coalitions: Optimality and stability. Econometrica 48, 987–1003 (1980)
7. Flake, G., Tarjan, R., Tsioutsiouliklis, K.: Graph clustering and minimum cut trees. Internet Mathematics 1(4), 385–408 (2004)
8. Greenberg, J.: Coalition structures. In: Aumann, R.J., Hart, S. (eds.) Handbook of Game Theory, vol. 2. Elsevier Science, B.V, Amsterdam (1994)
9. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)
10. Karger, D., Stein, C.: An $\widetilde{O}(n^2)$ algorithm for minimum cuts. In: STOC (1993)

11. Keinan, A., Sandbank, B., Hilgetag, C., Meilijson, I., Ruppin, E.: Axiomatic scalable neurocontroller analysis via shapley value. Artificial Life 12 (2006)
12. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 77, 273–324 (1997)
13. Luxburg, U.: A tutorial on spectral clustering. Stat. and Comput. 17(4) (2007)
14. Meilă, M., Jordan, M.: Learning with mixtures of trees. JMLR 1, 1–48 (2000)
15. Myerson, R.: Game Theory: Analysis of Conflict. Harvard University Press, Cambridge (1997)
16. Olsen, M.: Nash stability in additively separable hedonic games is NP-hard. In: Cooper, S.B., Löwe, B., Sorbi, A. (eds.) CiE 2007. LNCS, vol. 4497, pp. 598–605. Springer, Heidelberg (2007)
17. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. on PAMI 27(8), 1226–1237 (2005)
18. Perkins, S., Lacker, K., Theiler, J.: Grafting: Fast, incremental feature selection by gradient descent in function space. JMLR 3, 1333–1356 (2003)
19. Stoer, M., Wagner, F.: A simple min-cut algorithm. J. of the ACM 44(4) (1997)
20. Vazirani, V.V.: Approximation Algorithms. Springer, Heidelberg (2004)

Feature Selection Strategy in Text Classification

Pui Cheong Gabriel Fung, Fred Morstatter, and Huan Liu

Arizona State University
Attn: Gabriel Fung
P.O. Box 878809
Tempe, AZ 85287-8809
United States of America
{Gabriel.Fung,Fred.Morstatter,Huan.Liu}@asu.edu

Abstract. Traditionally, the best number of features is determined by the so-called "rule of thumb", or by using a separate validation dataset. We can neither find any explanation why these lead to the best number nor do we have any formal feature selection model to obtain this number. In this paper, we conduct an in-depth empirical analysis and argue that simply selecting the features with the highest scores may not be the best strategy. A highest scores approach will turn many documents into zero length, so that they cannot contribute to the training process. Accordingly, we formulate the feature selection process as a dual objective optimization problem, and identify the best number of features for each document automatically. Extensive experiments are conducted to verify our claims. The encouraging results indicate our proposed framework is effective.

Keywords: Feature Selection, Feature Ranking, Text Classification, Selection Strategy.

1 Introduction

Feature selection is used to control the dimensionality of the feature space so as to eliminate noise from the feature set and reduce the computational complexity. In the text domain, the number of features is on the order of ten thousand. Feature selection is a must in most cases. We use the popular vector space model for text representation. Every text document is regarded as a vector, and every vector consists of a set of words. Every word is regarded as a feature. As a result, we have a multi-dimensional space.

Broadly speaking, feature selection can be divided into two steps: (1) rank all features in the corpus by using some scoring function, and (2) select some of the top-ranked features, such that any feature in the corpus which does not belong to those selected features would be disregarded. The selected features are usually either the top K or top K percent of the ranked features. While we agree that there are already numerous effective scoring functions for feature selection ([1], [2], [3], [4], [6], [7], [8], [9], [13], [14], [15], [17], [18], [19]), we argue that we do not have any systematic approach for determining K. Identifying an optimal K is always treated as an empirical question by training and evaluating the classifier for a range of different K using a separate validation dataset or simply by the so-called "rule of thumb". Usually we have no understanding about the

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 26–37, 2011.

properties of the selected features and do not know why they perform the best in that particular K.

In addition, we have found that simply selecting the top K features may not always lead to the best classification performance. In fact, it may turn many documents into zero length, and they cannot contribute to the classifier training. This observation has not been recorded elsewhere to the best of our knowledge. In this paper, we formulate the feature selection process as a dual objective optimization problem: maximize the information presented in the documents and minimize the feature dimensionality. It is worthwhile to note that in our formulation, a feature which is being regarded as useful and retained in one document may not necessarily be retained in another document as well. Details will be discussed in later sections.

In this paper we will provide an in-depth analysis for the existing feature selection strategy, point out its fallacies and limitations, and suggest a novel model for selecting the features automatically. All discussion of feature selection in this paper will be with regards to the text domain. This work will also provide a documented framework for conducting feature selection in text classification such that the classifier performances would be optimized. By doing so, we can have a fairer comparison among different classification algorithms. For instance, Naive Bayes is long been regarded as inferior to Support Vectors Machine. Yet, how inferior is it? By fine tuning the text preprocessing, we will see that their differences are lower than expected.

The rest of this paper is organized as follows. Section 2 presents the issues related to feature selection; Section 4 discusses existing works; Section 3 reports the experimental results; Section 5 concludes this paper.

2 Feature Selection

For research that focuses on feature selection, the researchers merely aim at studying how effective a scoring function (e.g. Information Gain, χ^2, mutual information, etc.) is. In this paper, we move the research in this line a step forward by not only relating the classifier performances on the number of features that have to be selected, but also attempt to answer the question of why such a number would be optimal.

2.1 An Overview

A typical feature selection framework usually contains the following three steps:

1. **Feature Scoring.** A text corpus usually contains many text categories. Let C be a set of all categories in the corpus, and $c_k \in C$ be a category in C. In each c_k, we use a pre-defined scoring function (e.g. Information Gain) to assign scores to the features based on their distributions in c_k. Since we have $|C|$ number of categories, as a result, each feature has $|C|$ number of different scores. Some of the most widely used and well-documented scoring functions are: Information Gain [1,7,8,19,18], χ^2 [1,4,15,17], NGL Coefficient [13], GSS Coefficient [4], odd ratio [1,14], DIA association factor [3], and inverse document frequency [16]. These functions inherit the idea that the best features for c_k are those which are distributed most differently between c_k and $C - c_k$. Yet, interpreting this idea varies across different scoring functions, and this results in different kinds of computation.

2. **Feature Ranking.** In the previous step, we have assigned $|C|$ number of different scores to each feature. Let s_{jk} be the score of feature f_j in c_k. To determine which of the features are the dominant ones, for each feature, f_j, we have to combine its scores from all c_k to obtain a single value to denote its overall importance. Let ν_j be such a combined value. The most widely used combination functions are [16]: (1) linear summation $\nu_j = \sum_k s_{jk}$); (2) Weighted sum ($\nu_j = \sum_k P(c_k)s_{jk}$); (3) Maximum ($\nu_j = \max_k s_{jk}$).
3. **Feature Selection.** According to the overall scores of the features in the previous step, we select the features with the top K highest scores and regard them as useful indicators for classification. Any feature that does not belong to these top K features is removed from the corpus. To our knowledge, all existing research treats the issue of identifying an optimal K as an empirical question. It does this by training and evaluating the classifier for a range of different K using a separate validation dataset.

2.2 Analysis

Figure 1(a) shows the accuracy of a multi-nominal Naive Bayes classifier [11] versus the number of features selected.[1] The x-axis denotes the number of features selected and the y-axis denotes the accuracy of the classifier. The accuracy is measured using the well-known $F1$-measure [16]. Each line in the figure denotes one of the following commonly used scoring functions: (1) Information Gain (*IG*); (2) χ^2; (3) Odds ratio (*OR*); (4) GSS Coefficient (*GSS*); and (5) Inverse document frequency (*IDF*). Naive Bayes is chosen because it does not require feature weighting, and therefore its performance depends solely on the number of features selected.

In Figure 1(a), we can see that different scoring functions have different optimal values. If we want to understand "*why*", reviewing their mathematical formulas may not help since they *all adopt the same idea* that the best features for c_k are those distributed most differently between c_k and $\bar{c}_k$. Hence, we need further analysis in order to gain a better understanding.

Logically, when K is small (few features have been selected), the number of features remaining in a document will be even fewer. Eventually, a document may contain no features (i.e. its size is zero) if none of its features are found in the top K features. The number of training examples will shrink if the number of zero-size documents increases. Intuitively, the number of documents with zero size will affect the classification accuracy negatively because these documents cannot contribute to the classifier training process (since there are no features in these documents). Figure 1(b) plots the number of features selected against the percentages of documents with non-zero size. In the figure, the x-axis denotes the number of features selected and the y-axis denotes the percentage of non-zero-size documents (i.e. contains at least one feature).

In Figure 1(b), we can see that only *IDF* will assign higher scores to features that appear in only a few documents across the corpus. We understand that *IDF* is not a good feature selection function for text classification [16] [18] [19], but for the purposes of this illustration we try to include it. We will explain why later in the paper. In this

[1] Details of the settings will be given in Section 3.

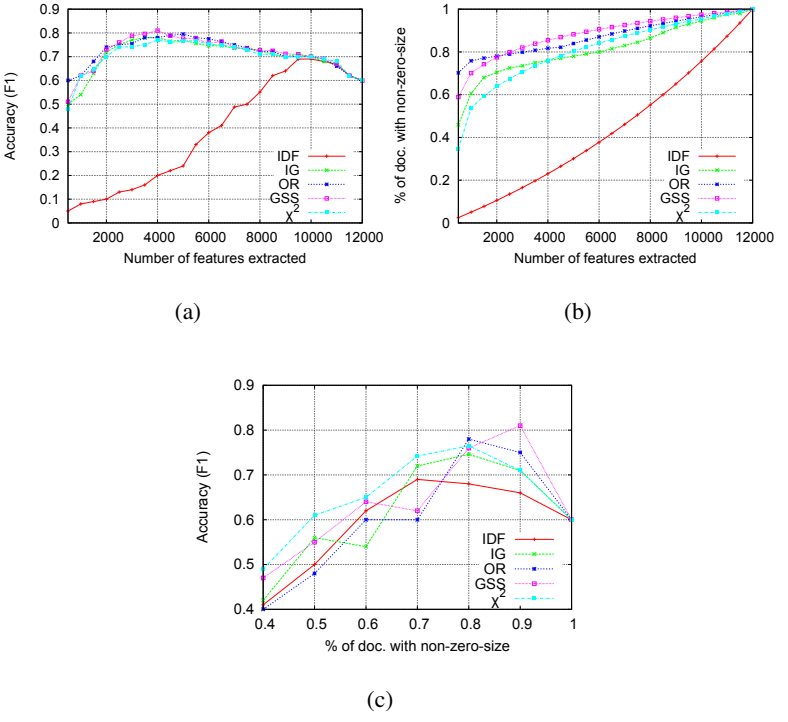
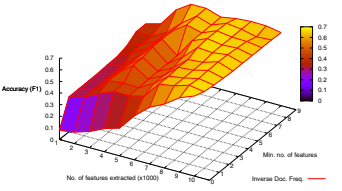


Fig. 1. Analysis of the feature distribution. (a) No. of features selected vs. Accuracy; (b) No. of features selected vs. Percentage of documents with non-zero size; (c) Percentage of documents with non-zero size vs. Accuracy.

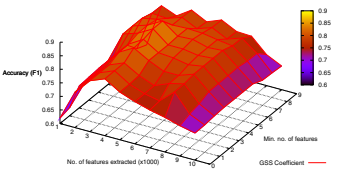
case, it will yield a very sparse document representation. According to Figure 1(a) the effectiveness of the scoring functions is roughly: $GSS > OR > IG > \chi^2 > IDF$, where the number of non-zero-size documents (according to Figure 1(b)) is also of this order. Thus, we believe that there may be a relationship between the number of non-zero-size documents remaining after feature selection and the performance of the classifier.

In order to have a better understanding of the relationship between the number of zero-size documents and classifier performance, we plot a graph with the percentage of non-zero-size documents against $F1$-measure in Figure 1(c). In this figure, we found that the effectiveness among different scoring functions seems to be "improved" (and much similar) with respect to Figure 1(a).

Until now, it should be clear that different scoring functions will affect the distribution of zero-size documents greatly. Yet, it does not mean including these documents in training could improve the classification accuracy. So, we conduct another analysis as follows. For each document, we try to maintain it to have at least M features. If the number of features remaining in the document is less than M after feature selection, then we will retain those features that are rank highest in that document until the size of the document is M. Figure 2 shows two plots of this analysis by using *GSS* (overall the best scoring function) and *IDF* (overall the worst scoring function). In the figures,



(a) Inverse Document Frequency (*IDF*)



(b) GSS Coefficient (*GSS*)

Fig. 2. The relationship among three elements: (1) No. of features selected (x-axis); (2) Min. no. of features has to be reminded in a document (y-axis); (3) Accuracy (z-axis)

x-axes are the number of features selected (in the order of 1000), y-axes are the value of M, and z-axes is the $F1$-measure. For the purpose of visualization, we contrain $M < 10$. Setting a different M does not affect the conclusions of this paper.

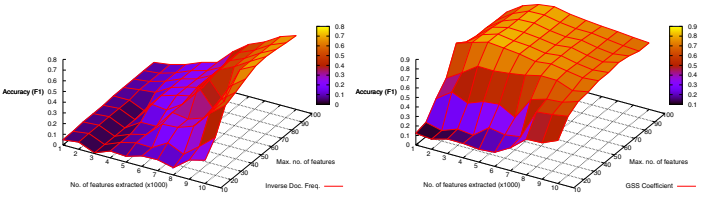
Regardless of the scoring functions, both of them in Figure 2 show a trend that when M increases to some value, the accuracy of the classifier will be maximum given that the number of features selected is the same. This is especially obvious for the scoring function *IDF* which increases performance from 0.08 to 0.389 when $M = 10$ and the number of features selected is 1000. This number is chosen as an example, setting any other value will arrive at the same conclusion. This is one of the major reasons why we include IDF in this analysis, because we can see that even a feature selection algorithm that is sub-optimal for our purposes could improve dramatically if we have an appropriate scheme for conducting feature selection. The best case for *GSS* here is 0.838 ($M = 8$ and 3000 features are selected), but it only obtains 0.810 if $M = 0$. These two figures suggest that those documents that would become zero-size after feature selection could in fact positively contribute to the classifier training *if we have some way to make them become "not zero-size"*.

To take a step forward, one may think that the number of features remaining in the document is highly related to the classifier effectiveness. In order to justify this statement, we conduct a final analysis. Identify the $F1$-measures of the classifier such that the maximum number of features in a document could not exceed M'. Figure 3 shows two plots for this analysis. Its definitions are similar to Figure 2 except that the z-axis becomes M' (the maximum number of features allowed in a document). Similar to Figure 2, their performances would be a maximum for some values of M' with a given number of features selected.

According to the analysis from Figure 1 to Figure 3, we claim that *it is not the number of features selected in the corpus that would affect the performances of the classifiers, but it is the number of features remaining in the documents*.

2.3 Modeling

We should not aim at identifying a single K that optimizes the performance over all categories. It will only lead to local optimization. We suggest that we should study at the document level – minimize the number of zero-size documents by preserving maximum information in a document. Yet, this objective alone is not enough, as one



(a) Inverse Document Frequency (*IDF*)

(b) GSS Coefficient (*GSS*)

Fig. 3. The relationship among three elements: (1) No. of features selected (x-axis); (2) Max. no. of features has to be reminded in a document (y-axis); (3) Accuracy (z-axis)

of the objectives of feature selection is to reduce the feature dimensionality. Thus, we have our second objective – minimize the number of features in the corpus. Eventually, a dual objectives optimization problem is formulated:

$$\max: \quad \sum_{\forall i}\sum_{\forall j} s_{jk}x_{ij}, \tag{1}$$

$$\min: \quad \sum_{\forall j} a_j y_j, \tag{2}$$

$$\text{subject to:} \quad x_{ij} = \{0,1\} \text{ and } y_j = \{0,1\}, \tag{3}$$

$$y_j = 1, \text{if } \exists x_{ij} = 1 \text{ for any } i, \tag{4}$$

$$y_j = 0 \text{ otherwise} \tag{5}$$

$$i \in [1,N] \tag{6}$$

$$j \in [1,M] \tag{7}$$

$$a_j = \frac{N}{\sum_i x_{ij}} \tag{8}$$

$$\text{where: } N \text{ is the number of documents and } M \text{ is the number of features} \tag{9}$$

where $x_{ij} = 1$ if the feature f_j should be retained in document d_i, and $x_{ij} = 0$ otherwise or f_j does not appear in d_i; s_{jk} is the score of f_j in category c_k computed by some function. The first objective (Eq. (1)) tries to maximize the information contained in every single document by retaining as many features as possible, while the second objective (Eq (2)) tries to minimize the number of features selected. Specifically, if feature f_j is selected in a document, y_j will be 1. Alternatively, if f_j is not selected in a document, y_j will be 0. a_j is a parameter trying to average the number of documents selected in the corresponding feature f_j. Our aim is to find the best combination of 0 and 1 for x, i.e. this is a 0-1 integer programming problem. Since multi-objective problems are difficult to solve, we combine Eq. (1) and Eq. (2):

$$\min: \quad \lambda\sum_{\forall j} y_j - \sum_{\forall i}\sum_{\forall j} s_{ij}x_{ij}, \tag{10}$$

where λ is the parameter to balance the two objectives. It has to be set carefully so as to balance the two components, as the first and second component will be heavily affected by the number of features and the number of documents, respectively. In this paper, we let λ be the ratio between these two factors by using a validation dataset. We plan to have more detail on finding the optimal λ in future works. We will show how effective this is in formulation of our experimental study.

3 Experiment

Extensive experiments are conducted using two benchmarks: Reuters-21578[2] and Newsgroup-20[3]. For Reuters-21578 we use ModApte spit and remove the categories that only appear in the training set or testing set. For Newsgroup-20 there are 20 different categories. For each category, we randomly select 80% of the postings as training and the remaining 20% as testing. We repeat the experiments 30 times and report the averaged result.

For each document, we remove any punctuation, numbers, Web page addresses, and email addresses. All features are stemmed using the Lovins stemmer [10]. Following the existing works of [16], features that appear in less than 1% of the documents are regarded as noise. Features that appear in more than 95% of the documents are regarded as stopwords. For feature weighting, we implemented the well-known *tf*·*idf* schema whenever necessary.

3.1 Algorithms for Comparison

For evaluating the proposed feature selection framework, we implemented all of the seven most widely used feature scoring functions as shown in Table 1, and compare our framework with the existing feature selection approach. The existing approach is to select only the top K by using a training and validation dataset such that this K would optimize the performance of the classifier on the validation dataset.

For the classification algorithm, we implemented the Naive Bayes [11] algorithm. Naive Bayes is chosen because it does not require feature weighting, so we can see the impact of different feature selection frameworks easily. We use the multinomial version of Naive Bayes and use Laplacian smoothing with $m = 0.01$ [11]. We have tried different values of m, and $m = 0.01$ results in the best performance for almost all situations. For performance evaluation we report only the micro-F1 value due to the space limitation. Yet, the trend of all other measurements (precision, recall, break-even point, etc.) have the same general pattern and can lead to the same conclusion.

3.2 Result and Discussion

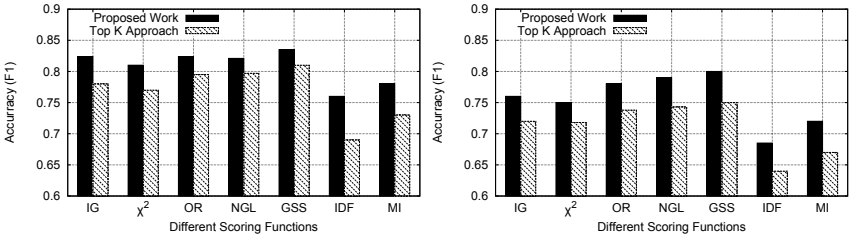
Figure 4 shows the classification results of Naive Bayes with the seven different scoring functions that are described in Section 4. Since *NB* does not require feature weighting when generating its model, we do not need to compare its performance with different

[2] www.daviddlewis.com/resources/testcollections/reuters21578/

[3] http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 1. Common scoring functions in feature selection

Weighting Schema	Mathematical Form	Citation
Information Gain (*IG*)	$\sum_{c\in\{c_i,\bar{c}_i\}}\sum_{f\in\{f_j,\bar{f}_j\}} P(f,c)\log\frac{P(f,c)}{P(f)P(c)}$	[1,7,8,19,18]
χ^2	$N\cdot\frac{[P(f,c)\cdot P(\bar{f},c)\cdot P(f,\bar{c})\cdot P(\bar{f},\bar{c})]^2}{P(f)\cdot P(\bar{f})\cdot P(c)\cdot P(\bar{c})}$	[1,4,15,17]
Odd ratio (*OR*)	$\frac{P(f\|c)\cdot(1-P(f\|\bar{c}))}{(1-P(f\|c))\cdot P(f\|\bar{c})}$	[1,14]
NGL coefficient (*NGL*)	$\sqrt{N}\cdot\frac{P(f,c)\cdot P(\bar{f},c)\cdot P(f,\bar{c})\cdot P(\bar{f},\bar{c})}{\sqrt{P(f)\cdot P(\bar{f})\cdot P(c)\cdot P(\bar{c})}}$	[13]
GSS coefficient (*GSS*)	$P(f,c)\cdot P(\bar{f},c)\cdot P(f,\bar{c})\cdot P(\bar{f},\bar{c})$	[4]
Inverse doc. freq. (*IDF*)	$\log\frac{N}{m(f,c)}$	[16]
Mutual information (*MI*)	$\log\frac{P(f,c)}{P(f)\cdot P(c)}$	[19]



(a) *Reuters21578* (b) *Newsgroup20*

Fig. 4. Classification result of *NB*

weighting schemes. In the figure, the black charts represent the results that are achieved by our proposed feature selection framework, while the shaded charts represent the results that are obtained by identifying one single K that maximizes the performance of the classifier on the validation dataset. We call this the "top K approach".

In the figures, regardless of which scoring function, all black charts (the proposed feature selection framework) dominate the shaded charts. This result indicates our proposed framework is highly effective. The improvement is especially visible for *MI* and *IDF*, which are the worst two among all seven scoring functions. If the total number of features selected is small, there will be many zero-size documents. On the other hand, if the total number of features select is too high, then the noise in the corpus will be high, which will eventually deteriorate the classifier performance as well. Our proposed feature selection framework could properly remedy the above two situations by minimizing the number of zero-size documents and minimizing the feature dimensionality.

Figure 5 shows the performance of Naive Bayes versus Support Vector Machines (SVM). SVM has long been cited for its high effectiveness and reliability when in conducting text classification. Note that SVM does not require feature selection for document preprocessing as it is designed for handling very high dimensionality in a very

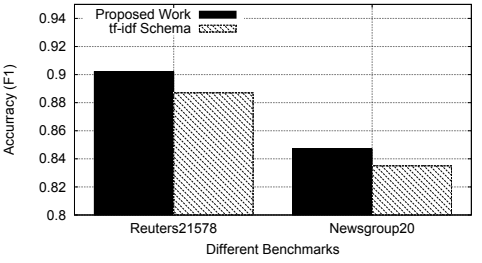
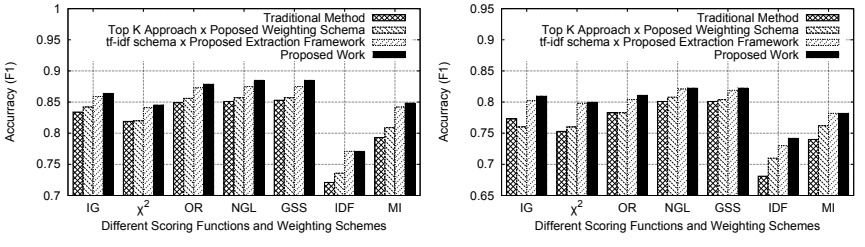


Fig. 5. Classification results of *SVM*

noisy environment. Yet SVM does require feature weighting. In this experiment, we use the traditional *tf·idf* scheme for feature weighting [18]. We use a linear kernel with $C = 1.0$ which is a common setting for SVM. In Figure 5, we can see that although the performance of SVM is still better than that of Naive Bayes, the difference is marginal. From [16], among the existing work on Naive Bayes in the Reuters dataset, the best performance is 0.795 whereas it is around 0.870 for SVM. However, from our study we have found that Naive Bayes could achieve around 0.84 for the Reuters dataset if we select the features properly. Similarly, for the Newsgroup dataset, we found that the performance could obtain a much higher accuracy than the reported studies using our framework.



(a) *Reuters21578* (b) *Newsgroup20*

Fig. 6. Classification result of *ROC*

4 Related Work

Feature selection is a topic which has permeated fields from statistics, to image processing. Largely, the goal of feature selection is to reduce dimensionality in data by selecting only the most relevant criterion for classifying a body. While studying feature selection, different researchers take different approaches to which attributes of the feature selection process are most important to the success of the algorihtm.

Some works focus on the scoring function as the way to identify the most important features. Yang and Pedersen [19] performed an in-depth study of feature selection in the text domain where they compared common scoring functions in feature selection.

The results of their study show that Information Gain [1,7,8,19,18] and χ^2 [1,4,15,17] are the most effective scoring functions for text classification.

In a survey conducted by Yu and Liu [9], the feature selection algorithms are classified into several clear and defined categories based upon their search strategies, evaluation criteria, and data mining tasks (the evaluation step for determining the validity of a subset). The result of this survey does not necessarily favor one of their defined criteria, however they propose a framework where the best feature selection algorithm is chosen behind the scenes, completely transparent to the user.

A technical report on feature selection carried out by Arizona State University [19] breaks feature selection algorithms down into four categories: if they are supervised, whether they are of filter or embedded model, whether they have univariate or multivariate variable selection, and whether they do weight selection or set selection. The results on the site largely suggest that no one category can determine the success of a feature selection algorithm.

This technical report breaks the feature selection process into four parts: feature subset generation, subset evaluation, stopping criteria, and results validation. The four steps are ordered nicely in Figure 7. While all of the mentioned papers present different criteria for what is most important to feature selection, this paper focuses on the "evaluation" step (as shown in Figure 7). As we have shown, having a robust evaluation framework can be crucial to the success of a feature selection algorithm.

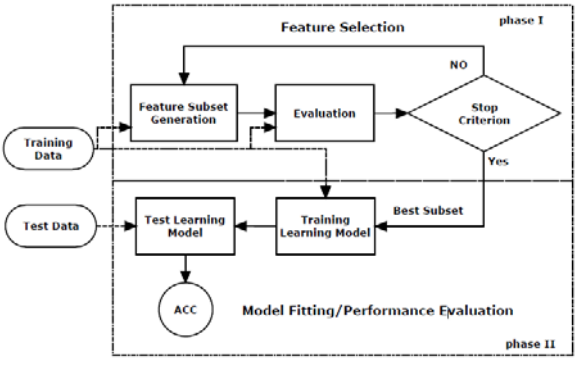


Fig. 7. Feature selection process as outlined by [19]. Figure used with permission.

5 Summary and Conclusion

In this paper, we study feature selection in the text preprocessing domain. We provide an in-depth analysis for the existing selection framework, point out its fallacies and limitations, and suggest a novel model for selecting the features automatically. We formulate the feature selection process as a dual objective optimization problem, and identify the best number of features for each document, rather than determining a fixed threshold which optimizes the overall classification accuracy for different categories. We provide

a documented framework for conducting text preprocessing in text classification in order to optimize the classifier performances, regardless of which classification model one intends to use. Extensive experiments are conducted to validate our claims. The favorable experimental results indicate that our proposed work is highly feasible.

References

1. Caropreso, M.F., Matwin, S., Sebastiani, F.: A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Chin, A.G. (ed.) Text Databases and Document Management: Theory and Practice, pp. 78–102. Idea Group Publishing, USA (2001)
2. Dumais, S.T., Platt, J.C., Hecherman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: Proceedings of the 1998 ACM CIKM International Conference on Information and Knowledge Management (CIKM 1998) (1998)
3. Futr, N., Hartmann, S., Knorz, G., Lustig, G., Schwanter, M., Tzeras, K.: AIR/X – a rule-based multistage indexing system for large subject fields. In: Proceedings of the 3rd International Conference on Intelligent Text and Image Handling (RIAO 1991) (1991)
4. Galavotti, L., Sebastiani, F., Simi, M.: Experiments on the use of feature selection and negative evidence in automated text categorization. In: Borbinha, J.L., Baker, T. (eds.) ECDL 2000. LNCS, vol. 1923, p. 59. Springer, Heidelberg (2000)
5. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 491–502. Springer, Heidelberg (1998)
6. Lam, W., Lai, K.Y.: A meta-learning approach for text categorization. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001) (2001)
7. Larkey, L.S., Croft, W.B.: Combining classifiers in text categorization. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1996) (1996)
8. Lewis, D.D.: An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1992) (1992)
9. Li, Y.H., Jain, A.K.: Classification of text documents. The Computer Journal 41(8), 537–546 (1998)
10. Lovins, J.B.: Development of a stemming algorithm. Mechanical Traqnslation and Computational Linguistics 11, 22–31 (1968)
11. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classification. In: The 15th National Conference on Artificial Intelligence (AAAI 1998) Workshop on Learning for Text Categorization (1998)
12. Moschitti, A.: A study on optimal parameter tuning for rocchio text classifier. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 420–435. Springer, Heidelberg (2003)
13. Ng, H.T., Goh, W.B., Low, K.L.: Feature selection, perception learning, and a usability case study for text categorization. In: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1997) (1997)
14. Ruiz, M.E., Srinivasan, P.: Hierarchical neural networks for text categorization. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999) (1999)

15. Schütze, H., Hull, D.A., Pedersen, J.O.: A comparison of classifiers and document representations for the routing problem. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1995) (1995)
16. Seabastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
17. Sebastiani, F., Sperduti, A., Valdambrini, N.: An improved boosting algorithm and its application to automated text categorization. In: Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management (CIKM 2000) (2000)
18. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999) (1999)
19. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proceedings of the 14th International Conference on Machine Learning (ICML 1997) (1997)

Unsupervised Feature Weighting Based on Local Feature Relatedness

Jiali Yun, Liping Jing*, Jian Yu, and Houkuan Huang

School of Computer and Information Technology,
Beijing Jiaotong University, China
{yunjialiyjl,lpjinghk}@gmail.com,
{jianyu,hkhuang}@bjtu.edu.cn

Abstract. Feature weighting plays an important role in text clustering. Traditional feature weighting is determined by the syntactic relationship between feature and document (e.g. TF-IDF). In this paper, a semantically enriched feature weighting approach is proposed by introducing the semantic relationship between feature and document, which is implemented by taking account of the local feature relatedness — the relatedness between feature and its contextual features within each individual document. Feature relatedness is measured by two methods, document collection-based implicit relatedness measure and Wikipedia link-based explicit relatedness measure. Experimental results on benchmark data sets show that the new feature weighting approach surpasses traditional syntactic feature weighting. Moreover, clustering quality can be further improved by linearly combining the syntactic and semantic factors. The new feature weighting approach is also compared with two existing feature relatedness-based approaches which consider the global feature relatedness (feature relatedness in the entire feature space) and the inter-document feature relatedness (feature relatedness between different documents) respectively. In the experiments, the new feature weighting approach outperforms these two related work in clustering quality and costs much less computational complexity.

Keywords: Text Clustering, Feature Weighting, Feature Relatedness, Semantics.

1 Introduction

Text clustering is one of the most common themes in data mining field. As we know, text data is unstructured, which cannot be directly processed by the clustering algorithms. Thus text representation plays a vital role in text mining. Vector space model (VSM) borrowed from information retrieval is popularly used here. In VSM, a document is represented as a feature vector which consists of the words, phrases or other semantic units to identify the contents of the document. Different features have different weights according to their importance for the specific mining task. There are two open problems in VSM. One is how to select

* Corresponding author.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 38–49, 2011.

appropriate features. The other is how to weight these features. In this paper, we focus on solving the second problem, feature weighting.

Feature weighting approaches can be divided into two groups according to whether the approach makes use of the prior category label information, i.e. supervised feature weighting and unsupervised feature weighting [1]. In this paper, we aim to improve unsupervised feature weighting because our task is text clustering, where there is no known membership information in the data set.

Traditional feature weighting approaches (e.g. TF-IDF) directly compute the syntactic relationship between feature and document via some statistical methods. Generally, it consists of two factors: feature frequency factor representing the content of a document and document frequency factor indicating the feature's discriminating power [2]. However, they consider neither the semantic relationship between feature and document nor the relatedness between features.

To enrich text representation with semantic information, researchers have begun to take advantage of feature relatedness for re-weighting feature [3,4] and computing document similarity [5]. Jing et al. [3] and Wang et al. [4] embedded the semantic information to document representation by multiplying document-feature tf-idf matrix and feature relatedness matrix. Their new feature weight contains all the relatedness between each pair of features in the whole document collection. Huang et al. [5] designed a new document similarity measure by taking account of the feature relatedness between two documents. However, both these two approaches need to cost too much computational complexity.

In this paper, we propose a new semantically enriched feature weighting method by taking account of the local feature relatedness — the relatedness between features within each individual document. In other words, the weight of each feature in document is calculated according to the relatedness between the feature and its contextual content. Here, the features in the same document are taken as the contextual content. In this case, we can say the local feature relatedness weight implies the semantic relationship between feature and document because the whole features of each document can express the document's topic. By this method, feature highly semantically related to document's topic can be arranged with a higher weight. In addition, it takes much less computational complexity than the above two feature relatedness-based approaches.

In order to measure the feature relatedness, we adopt two methods, document collection-based implicit relatedness measure and Wikipedia link-based explicit relatedness measure. The former mines feature relatedness on the basis of feature distribution in document collection, while the later explicitly calculates semantic relatedness between features according to Wikipedia hyperlink structure.

The rest of the paper is organized as follows: Section 2 describes document representation and two feature relatedness measure methods. In Section 3, we propose the local feature relatedness-based weighting approach and combine syntactic and semantic factors in terms of document similarity. Section 4 compares the new feature weighting method with two existing feature relatedness-based work. Section 5 describes the experimental methodology and discusses the experimental results. Finally, we conclude the paper in Section 6.

2 Preliminary

2.1 Document Representation

The original text documents cannot be directly processed by clustering algorithms. Therefore, documents need to be converted into a more manageable representation. Typically, a document is represented as a vector in feature space — a sequence of features [6]. The most representation models use terms as the features. Terms can be words or n-grams appearing in document.

With the development of the semantics-based text representation, concepts from background knowledge are also used as features in some literature. For example, terms appearing in document are first mapped to their most related concepts indicated by the Wikipedia articles via the hyperlink structure in Wikipedia, as mentioned in [7], and then document representation is built by replacing the terms with those concepts. Concept VSM has been wide used in existing semantics-based text mining [4,5,8,9,10].

In this paper, we propose the weighting approach by regarding both terms and concepts as features separately.

2.2 Relatedness Measure

Our purpose is to improve feature weighting with the aid of the relatedness between features. However, how to measure the feature relatedness is an open problem. In this section, we introduce two relatedness measures which will be used to improve feature weighting respectively later.

- **Implicit Relatedness Measure:** Feature relatedness is mined only from document collection itself without any background knowledge, thus we called it implicit relatedness measure. Firstly, each feature is represented as a document vector. The weight of each component is the tf-idf value of the feature in the document. Next, the relatedness between two features is defined as two corresponding vectors' cosine measure.
- **Explicit Relatedness Measure:** Semantic relatedness information is explicitly obtained from background knowledge (i.e. Wikipedia) rather than document collection itself, we call it explicit relatedness measure.

  If feature is represented as term, term need to be firstly mapped to the most related Wikipedia article (concept) according to [7,11]. After getting term's relevant Wikipedia article, the problem that measuring the relatedness between terms is equal to measuring the relatedness between Wikipedia articles.

  To calculate semantic relatedness between Wikipedia articles, We adopt the Wikipedia link-based semantic relatedness measure proposed in [12]. Wikipedia link-based measure is simple because it ignores Wikipedia's extensive textual content, but more accurate because it is more closely tied to the manually defined semantics of the resource. It includes two aspects: one is based on the links extending out of each article, the other is based on the links made to them. The two measures are combined by addition operation.

The measure based on outgoing links is defined by the angle (cosine measure) between the vectors of the links found within two articles. The weight of each component in the vector depends on the probability of the corresponding link. By the link probability, links are considered less significant for judging the relatedness between articles if many other articles also link to the same target. In other words, link to general concept is much less significant than link to a specific topic [12].

The measure based on incoming links is based on link occurrences on Wikipedia pages. Pages that contain both links indicate relatedness, while pages with only one of the links suggest the opposite.

3 Feature Weighting

3.1 Feature Weighting Based on Syntactic Information

Feature weighting is usually implemented on the basis of syntactic relationship between feature and document via some statistical methods, inspired by information retrieval field. Salton et al. [2] discussed several considerations. First, because terms that are frequently mentioned in an individual document appear to be useful to represent the content of the document, terms frequency factor (e.g. normal raw Term Frequency, TF) is always used as part of the feature weighting system. Second, collection-dependent factor (e.g. Inverse Document Frequency, IDF) is introduced to increase the term's discriminating power to pick up all the relevant documents from other irrelevant document. In general, these two factors are combined by a multiplication operation.

The most popular syntactic feature weighting strategy is TF-IDF [13] showed by Eq. (1). Here, $tf(d_j, t_i)$ is the frequency of term t_i appears in document d_j. The IDF factor varies inversely with the number $df(t_i)$ of documents which contain the term t_i in a collection of N documents.

$$W_{syn}(d_j, t_i) = W_{TFIDF}(d_j, t_i) = tf(d_j, t_i) \times log(\frac{N}{df(t_i)}) \tag{1}$$

TF-IDF indicates the syntactic relationship between feature and document, which is determined by taking just lexical information into account. It can not reflect the semantic relationship between feature and document. Meanwhile, the weight of feature is independent on other features.

3.2 Feature Weighting Based on Local Feature Relatedness (LFR)

To design a new weight which can represent the semantic relationship between term and document, we propose a semantically enriched feature weighting approach by taking account of local feature relatedness — the feature relatedness within each individual document.

Human understand document's topic through the document's content — a sequence of terms, thus all the terms (selected features) in one document can express the document's meaning. The semantic relationship between term and

document can be calculated by the total semantic relatedness between the term and all the terms (called contextual terms) in the document.

As the contextual terms, their importance can be differentiate with the aid of syntactic information. In other words, contextual term with higher syntactic weight is more important to represent the document's content. Thus relatedness $Rel(t_k, t_i)$ of each pair of terms is weighted by the syntactic weight of the contextual term t_k in document d_j. The semantic feature weighting based on local feature relatedness is formulated as follows.

$$W_{sem}(d_j, t_i) = W_{LFR}(d_j, t_i) = \sum_{t_k \in T_{d_j}} W_{syn}(d_j, t_k) \times Rel(t_k, t_i) \tag{2}$$

where T_{d_j} means the term set of document d_j, $Rel(t_k, t_i)$ is the relatedness between term t_k and t_i introduced in Section 2.2, $W_{syn}(d_j, t_k)$ represents the syntactic weight of contextual term t_k in document d_j.

So far, we have given the new feature weighting approach. Based on this approach, term will be set to more higher weight if it is more semantically related to those important contextual terms in document. Here the importance of contextual terms for document's topic discrimination is measured by syntactic feature weighting approach (e.g. TFIDF).

Feature weighting based on local feature relatedness only needs to take $O(Nm^2)$, where N is the document number in the collection and m represents the average number of features in one document. Furthermore, in clustering task, the document similarity cosine measure takes $O(\frac{1}{2}N^2m + Nm)$. Here, $\frac{1}{2}N^2m$ is for computing the dot products of $\frac{1}{2}N^2$ pairs of document vectors (the numerator of Eq.(3)), and Nm is for computing the L2-norms of N document vectors (the denominator of Eq.(3)). It is more fast than two existing relatedness-based document representation approaches which will be described in next section.

3.3 Combination of Syntactic and Semantic Factors

In order to consider both syntactic and semantic information in text clustering, we combine these two factors on the basis of document similarity following [5]. Syntactic (semantic) document similarity is computed by cosine measure as showed in Eq.(3). $Sim(d_i, d_j) = Sim_{syn}(d_i, d_j)$ when $W(d, t) = W_{syn}(d, t)$, and $Sim(d_i, d_j) = Sim_{sem}(d_i, d_j)$ when $W(d, t) = W_{sem}(d, t)$

$$Sim(d_i, d_j) = \frac{\bar{d_i} \bullet \bar{d_j}}{||\bar{d_i}||||\bar{d_j}||} = \frac{\sum_{t_k \in \{T_{d_i} \cup T_{d_j}\}} W(d_i, t_k) \times W(d_j, t_k)}{\sqrt{\sum_{t_p \in T_{d_i}} W^2(d_i, t_p)} \sqrt{\sum_{t_q \in T_{d_j}} W^2(d_j, t_q)}} \tag{3}$$

The overall document similarity is defined as linear combination of syntactic similarity Sim_{syn} and semantic similarity Sim_{sem}:

$$Sim_{overall}(d_i, d_j) = \lambda Sim_{syn}(d_i, d_j) + (1 - \lambda) Sim_{sem}(d_i, d_j) \tag{4}$$

4 Related Work

To our knowledge, local feature relatedness has not been used for feature weighting. However, global feature relatedness and inter-document feature relatedness have appeared in the literature.

4.1 Feature Weighting Based on Global Feature Relatedness (GFR)

Different from our feature weighting based on local feature relatedness, some researchers improved feature weighting by global feature relatedness — the relatedness of all the feature pairs in the entire feature space. Jing et al [3] and Wang et al. [4] enriched semantic information into feature weight by multiplying document-term (concept) matrix by term (concept) correlation matrix. Their weight is calculated by Eq.(5).

$$W_{sem}(d_j, t_i) = W_{GFR}(d_j, t_i) = \sum_{t_k \in T_D} W_{syn}(d_j, t_k) \times Rel(t_k, t_i) \tag{5}$$

It is worth noting that T_D are the term set in the entire document collection. Although both GFR and LFR utilize syntactic weight value and the feature relatedness, GFR ignores the semantic relatedness within each individual document. This is main difference between these two weighting approaches.

GFR-based feature weighing is also a semantic feature weighting approach. In text clustering, it can be used to compute semantic document similarity, which is combined with syntactic document similarity accord to Eq.(3) and Eq.(4).

GFR needs to take $O(NMm)$, where M represents the number of features (terms or concepts) in the document collection. Because almost all the features in the document collection have non-zero weight in each document when using GFR-based feature weighting, the document similarity cosine measure takes $O(\frac{1}{2}N^2M + NM)$, which is different from $O(\frac{1}{2}N^2m + Nm)$ of LFR.

4.2 Document Similarity Based on Inter-document Feature Relatedness (IFR)

In the literature, inter-document feature relatedness (the relatedness between features from different documents) was not used for feature weighting, but document similarity measure. Huang et al. [5] represented document by a set of concepts, each with a syntactic weight. Semantic relatedness between concepts was applied to the similarity between documents which is defined as:

$$Sim_{sem}(d_i, d_j) = Sim_{IFR}(d_i, d_j) = \frac{\sum_{\forall c_k \in C_{d_i}, \forall c_l \in C_{d_j}} W_{syn}(d_i, c_k) \times W_{syn}(d_j, c_l) \times Rel(c_k, c_l)}{\sum_{\forall c_k \in C_{d_i}, \forall c_l \in C_{d_j}} W_{syn}(d_i, c_k) \times W_{syn}(d_j, c_l)} \tag{6}$$

Where C_{d_i} and C_{d_j} are the concept sets in document d_i and d_j respectively. $W_{syn}(d_i, c_k)$ means the syntactic weight of concept c_k in document d_i. $Rel(c_k, c_l)$ represents the semantic relatedness between two concepts c_k and c_l, it is equal to the corresponding Wikipedia articles' relatedness introduced in Section 2.2.

In contrast to inter-document feature relatedness, our local feature relatedness is also called intra-document feature relatedness.

Here, semantic document similarity is directly computed by Eq.(6). As presented in [5], it can be further combined with syntactic similarity by Eq.(4).

To compute the semantic similarity between each two documents, it has to take $O(\frac{2}{2}N^2m^2)$. The numerator 2 means two multiply operations are required in each loop body (see Eq. (6)).

5 Experiments

5.1 Datasets

The proposed weighting approach was tested on two real data, 20Newsgroups[1] and Reuters-21578[2]. We extracted two subsets from 20Newsgroups following [14]:

- **20NG-Multi5** was extracted from 5 categories (comp. graphics, rec. motorcycles, rec. sport. baseball, sci. space, talk. politics. mideast).
- **20NG-Multi10** was extracted from 10 categories (alt. atheism, comp. sys. mac. hardware, misc. forsale, rec. autos, rec. sport. hockey, sci. crypt, sci. electronics, sci. med, sci. space, talk. politics. guns).

Other two data subsets were created from Reuters-21578 following [5]:

- **R-Min20Max200** consists of 25 categories with at least 20 and at most 200 documents.
- **R-Top10** contains 10 largest categories in the original data set.

For efficiency, 20NG-Multi5, 20NG-Multi10 and R-Top10 were created by randomly picking 100 documents from each selected category. Table 1 lists the number of categories and documents contained in these subsets. In this paper, we only consider the single-label documents. Wikipedia is used as background knowledge base which contains 2,388,612 articles and 8,339,823 anchors in English.

In each data set, the terms were extracted by preprocessing steps, selecting only alphabetical sequences, stemming them, removing stop words and filtering them by the document frequency. The concepts were mapped from terms according to [7,11]. The number of terms and concepts extracted from each data set are showed in Table 1.

Table 1. Data set summary

Dataset	Classes	Documents	Terms	Concepts	Terms no Concepts
20NG-Multi5	5	500	3310	2735	302
20NG-Multi10	10	500	3344	2772	285
R-Min20Max200	25	1413	2904	2450	176
R-Top10	10	1000	1980	1720	100

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/
[2] http://kdd.ics.uci.edu/databases/reuters21578/

5.2 Methodology

In the experiments, we have two purposes. The first one is to compare the LFR-based feature weighting with existing work, including syntactic feature weighting, GFR-based feature weighting [3,4] and IFR-based document similarity measure [5] in clustering task. The second one is to validate whether the combination of syntactic and semantic factors can achieve better performance. The best combined result is reported by changing the parameter λ in Eq.(4) from 0.1 to 0.9, with 0.1 interval. Here, the syntactic weight is calculated by TFIDF. For simplicity, the abbreviations LFR, GFR and IFR are used to represent three feature relatedness-based approaches in the following parts.

We adopt Bisecting k-means (Bi-KM) algorithm which is proven to be the best document clustering technology [15]. In the experiments, Bi-KM in CLUTO[3] toolkit is used. The parameter of Bi-KM, the number of clusters, is set to be the true number of classes for each data set.

The input of Bi-KM is document similarity matrix. Different approaches tested in the experiments and their input matrices are described in Table 2.

Table 2. Methodology

Notation	Input
TFIDF	cosine similarity based on TFIDF feature weighting
GFR	cosine similarity based on GFR feature weighting
IFR	IFR similarity
LFR	cosine similarity based on LFR feature weighting
TFIDF+GFR	linear combination of TFIDF-based cosine similarity and GFR-based cosine similarity
TFIDF+IFR	linear combination of TFIDF-based cosine similarity and IFR similarity
TFIDF+LFR	linear combination of TFIDF-based cosine similarity and LFR-based cosine similarity

All the above approaches can be applied on Term VSM and Concept VSM. Meanwhile, the feature relatedness measure in GFR, LFR and IFR is calculated by two strategies: document collection-based implicit relatedness measure (Imp) and Wikipedia link-based explicit relatedness measure (Exp). Therefore, each approach in Table 2 is implemented in four schemas: Term+Imp, Concept+Imp, Term+Exp and Concept+Exp (In TFIDF, Term+Imp is equal to Term+Exp and Concept+Imp is equal to Concept+Exp, because TFIDF does not refer the feature relatedness measure.).

5.3 Evaluation Metrics

Since the class label of each document is known in data sets, we use the external cluster validation method to evaluate the clustering results by calculating the correspondence between the clusters generated by clustering algorithm and the inherent classes of the data. Cluster quality is evaluated by *F-Score* [16] and the normalized mutual information (*NMI*) [17]. F-score combines the information of precision and recall. NMI is defined as the mutual information between the

[3] http://glaros.dtc.umn.edu/gkhome/views/cluto

cluster assignments and a pre-existing labeling of the dataset normalized by the arithmetic mean of the maximum possible entropies of the empirical marginal. Both the two metrics range from 0 to 1, and the higher their value, the better the clustering quality is.

5.4 Experiment Results

Table 3 compares LFR with syntactic feature weighting (TFIDF) and other two feature relatedness-based approaches (GFR, IFR) in four schemas: Term+Imp, Concpet+Imp, Term+Exp and Concept+Exp. Table 4 shows the clustering results by combining syntactic factor with semantic factor in terms of document similarity. **Bold-face** number indicates the best result among different approaches. The " * " indicates the best result among different schemas.

From Table 3, we can see LFR achieves the better performance no matter which schema is adopted. The success of LFR is due to the use of the feature relatedness within each individual document. The relatedness between feature and its contextual features implies the semantic relationship between feature and document. IFR utilizes the semantic relatedness between features from different documents. LFR and IFR are designed from two different aspects, intra-document similarity and inter-document similarity. GFR updates feature weight by feature relatedness in the entire document collection. It neither implies the contextual information as LFR, nor implies the feature's discriminating power as IFR. Thus GFR is worse than both LFR and IFR. Moreover, GFR is more easy to introduce noise because all the feature pairs are referred.

Comparing with syntactic feature weighting, LFR always surpasses TFIDF but GFR and IFR fail in some cases. What is the degree that LFR improves TFIDF? This is partly due to the quality of feature relatedness. In the experiments, feature relatedness threshold is changed from 0 to 0.3, we found the smaller threshold value can usually achieve the better performance but at the cost of time. In this paper, we report the results when threshold is set to 0. In the future, we expect to find more proper feature relatedness measure to achieve much better semantic feature weighting.

Comparing Table 4 with Table 3, we can see the clustering results with the combined method are always better than using TFIDF or LRF (GRF, IRF) solely. In other words, text clustering can be further improved by combining syntactic factor with semantic factor, which is enriched by feature relatedness. From Table 4, we can also see that TFIDF+LRF is usually better than both TFIDF+GRF and TFIDF+IRF.

Comparing the implicit relatedness measure with the explicit relatedness measure, LFR with explicit schema surpasses implicit schema in most cases, while GFR and IFR with explicit schema is worse than implicit schema. This is an interest phenomenon. The explicit relatedness is calculated by mining the semantic information from Wikipedia, which contains abundant information from all kinds of domains. With aid of Wikipedia, two related terms (concepts) can get high relatedness indeed. However, terms (concepts) which are unrelated in the data set might have also relatedness in Wikipedia. In contrast, implicit relatedness is

Table 3. Comparison of clustering performances with TFIDF, GFR, IFR and LFR

Dataset	Approaches	Term+Imp F-Score	Term+Imp NMI	Concept+Imp F-Score	Concept+Imp NMI	Term+Exp F-Score	Term+Exp NMI	Concept+Exp F-Score	Concept+Exp NMI
20NG-Multi5	TFIDF	0.9440	0.8433	0.9641	0.8963	0.9440	0.8433	0.9641	0.8963
	GFR	0.9681	0.9103	0.9601	0.8842	0.6797	0.4648	0.7506	0.5448
	IFR	0.9681	0.9060	0.9621	0.8918	0.7999	0.6641	0.8064	0.6676
	LFR	**0.9860**	**0.9571**	**0.9880**	**0.9623**	**0.9960***	**0.9878***	**0.9940**	**0.9808**
20NG-Multi10	TFIDF	0.7745	0.6604	0.6634	0.5832	0.7745	0.6604	0.6634	0.5832
	GFR	0.6449	0.6246	0.6580	0.6240	0.4014	0.2857	0.4266	0.3165
	IFR	0.7166	0.6898	0.7316	0.7062	0.5949	0.4570	0.6241	0.4879
	LFR	**0.7786**	**0.6920**	**0.7472**	**0.7136**	**0.9038***	**0.8329***	**0.8619**	**0.7849**
R-Min20 Max200	TFIDF	0.6415	0.6656	0.6223	0.6537	0.6415	0.6656	0.6223	0.6537
	GFR	0.6035	0.6540	0.5698	0.6412	0.3445	0.3760	0.3692	0.3867
	IFR	0.6177	0.7048	0.6153	0.6969	0.4479	0.5528	0.5510	0.6102
	LFR	**0.6549**	**0.7143**	**0.6326**	**0.7161***	**0.6696***	**0.6837**	**0.6289**	**0.6937**
R-Top10	TFIDF	0.6685	0.6872	0.6910	0.6816	0.6685	0.6872	0.6910	0.6816
	GFR	0.5444	0.5750	0.5873	0.5989	0.3371	0.2479	0.3415	0.2400
	IFR	0.6797	0.6698	0.6700	0.6681	0.4679	0.4112	0.4849	0.4411
	LFR	**0.6933**	**0.6952**	**0.7229**	**0.7105**	**0.7352**	**0.7208**	**0.7661***	**0.7351***

Table 4. Comparison of clustering performances with combined methods (TFIDF+GFR, TFIDF+IFR and TFIDF+LFR)

Dataset	Approaches	Term+Imp F-Score	Term+Imp NMI	Concept+Imp F-Score	Concept+Imp NMI	Term+Exp F-Score	Term+Exp NMI	Concept+Exp F-Score	Concept+Exp NMI
20NG-Multi5	TFIDF+GFR	0.9820	0.9432	0.9660	0.9004	0.7997	0.7690	0.8301	0.7685
	TFIDF+IFR	0.9680	0.9015	0.9780	0.9337	0.9540	0.8673	0.9681	0.9060
	TFIDF+LFR	**0.9859**	**0.9571**	**0.9880**	**0.9651**	**0.9983***	**0.9930***	**0.9960**	**0.9878**
20NG-Multi10	TFIDF+GFR	0.7705	0.7020	0.7215	0.7055	0.6910	0.6261	0.7083	0.6626
	TFIDF+IFR	0.7895	0.6965	**0.7889**	0.7247	0.7700	0.7032	0.7833	0.7227
	TFIDF+LFR	**0.8299**	**0.7587**	0.7819	**0.7493**	**0.9039***	**0.8387***	**0.8755**	**0.8079**
R-Min20 Max200	TFIDF+GFR	0.6922	0.7692	0.7052	0.7792	0.7047	0.7738	0.7430	0.7754
	TFIDF+IFR	0.7165	**0.7929**	0.7298	0.7951	0.7316	0.7936	0.7412	0.8032
	TFIDF+LFR	**0.7166**	0.7809	**0.7311**	**0.8014**	**0.7406**	**0.8009**	**0.7433***	**0.8081***
R-Top10	TFIDF+GFR	0.6988	0.7166	0.7397	0.7267	0.6585	0.7069	0.7515	0.7421
	TFIDF+IFR	0.7536	0.7358	0.7271	0.7178	0.7598	0.7505	0.7528	0.7515
	TFIDF+LFR	**0.7739**	**0.7429**	**0.7507**	**0.7620***	**0.7740***	**0.7524**	**0.7675**	**0.7519**

computed according to the statistical information from document collection itself, which introduces less noise. From the experimental results, we conclude that LFR is more robust because it refers to less feature pairs than GFR and IFR. Meanwhile, it takes proper advantage of feature relatedness than GFR and IFR.

5.5 Computational Complexity

It is worth noticing that LFR is much faster than GFR and IFR. Given document-feature tf-idf weight matrix and feature relatedness matrix, we record the running time[4] of generating document similarity matrix using different approaches in Figure 1, taking Concept+Exp schema as an example. In Figure 1, IFR indicates the time of computing document similarity matrix using Eq. (6); while LFR (GFR) includes the time of constructing LFR (GFR)-based

[4] The time is recorded on the computer with Pentium(R) Dual-Core CPU E5400 @ 2.70GHz and 2GB RAM.

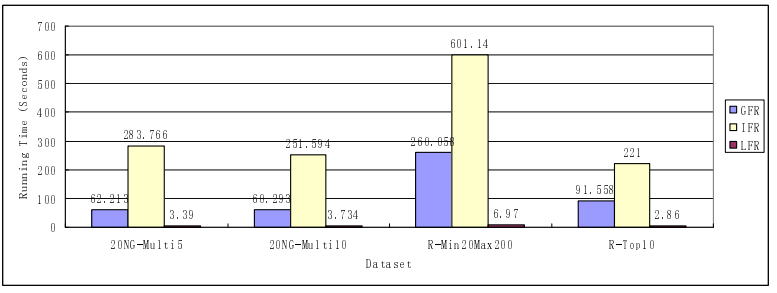


Fig. 1. Comparison of Running Time of GFR, IFR and LFR

document-feature weight matrix using Eq. (2) (Eq. (5)) and computing document similarity matrix using Eq.(3). We can see IFR is the most time-consuming among all the three approaches. This is because in IFR document semantic similarity refers to all the the feature pairs from different documents. GFR takes much more time than LFR. In GFR, the relatedness of all the feature pairs in the entire document collection are used to generate new document-feature weighting matrix which is very dense. In this process, too much multiply operation is required and dense matrix makes document similarity is slow to be computed. In contrast, LFR focuses on the feature relatedness within each individual document which greatly reduces the computational complexity. As mentioned earlier, the computational complexity of GFR, IFR and LFR is $O(NMm+\frac{1}{2}N^2M+NM)$, $O(\frac{2}{2}N^2m^2)$ and $O(Nm^2+\frac{1}{2}N^2m+Nm)$ respectively. In general, M is 10 to 100 times of m. The value of N and M is showed in Table 1. The obvious advantage in computational complexity makes our local feature relatedness-based feature weighting more useful in practice.

6 Conclusions

In this paper, we propose a semantic feature weighting approach by taking account of local feature relatedness — feature relatedness within documents. The feature relatedness can be computed by two strategies, document collection-based implicit relatedness measure and Wikipedia link-based explicit relatedness measure. From experimental results on real data, we conclude that: (1) The semantic feature weighting approach based on local feature relatedness surpasses the syntactic feature weighting (e.g. TFIDF); (2) The use of local feature relatedness surpasses global feature relatedness and inter-document feature relatedness in clustering quality and computational complexity; (3) Clustering results can be further improved by combining syntactic factor and semantic factor which is enriched by feature relatedness.

Acknowledgments. We thank Dr. David Milne for providing us with the Wikipedia Miner tool-kit. This work was supported by the Fundamental Research Funds for the Central Universities (2009YJS023, 2010RC029), the National Natural Science Foundation of China (60905028, 60905029, 90820013, 60875031,

61033013), 973 project (2007CB311002, 2007CB307100, 2007CB307106) and the Project-sponsored by SRF for ROCS, SEM.

References

1. Lan, M., Tan, C., Su, J.: Supervised and traditional term weighting methods for automatic text categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4), 721–735 (2009)
2. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
3. Jing, L., Zhou, L., Ng, M., Huang, J.: Ontology-based distance measure for text clustering. In: Proc. of the 4th Workshop on Text Mining, the 6th SIAM International Conference on Data Mining (2006)
4. Wang, P., Domeniconi, C.: Building semantic kernels for text classification using Wikipedia. In: Proc. of the 14th ACM SIGKDD, New York, NY, USA, pp. 713–721 (2008)
5. Huang, A., Milne, D., Frank, E., Witten, I.: Clustering documents using a wikipedia-based concept representation. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 628–636. Springer, Heidelberg (2009)
6. Feldman, R., Sanger, J.: The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press, Cambridge (2007)
7. Medelyan, O., Witten, I., Milne, D.: Topic indexing with wikipedia. In: Proc. of the AAAI Wikipedia and AI Workshop (2008)
8. Wang, P., Hu, J., Zeng, J., Chen, L., Chen, Z.: Improving text classification by using ecyclopedia knowledge. In: Proc. of the 7th ICDM, Omaha, NE, USA, pp. 332–341 (2007)
9. Huang, A., Milne, D., Frank, E., Witten, I.: Clustering documents with active learning using wikipedia. In: Proc. of International Conference on Data Mining Series, pp. 839–844 (2008)
10. Hu, X., Zhang, X., Lu, C., Park, E., Zhou, X.: Exploiting wikipedia as external knowledge for document clustering. In: Proc. of the 15th ACM SIGKDD, pp. 389–396 (2009)
11. Mihalcea, R., Csomai, A.: Wikify! linking documents to encyclopedic knowledge. In: Proc. of the 16th CIKM (2007)
12. Milne, D., Witten, I.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: Proc. of the Workshop on Wikipedia and Artificial Intelligence at AAAI, pp. 25–30 (2008)
13. Jones, K.S.: A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation 60(5) (2004)
14. Slonim, N., Tishby, N.: Document clustering using word clusters via the information bottleneck method. In: Proc. of 23th ACM SIGIR, pp. 208–215 (2000)
15. Steinbach, S., Karypis, G., Kumar, V.: A comparison of document clustering techniques. In: Proc. of the Workshop on Text Mining at ACM SIGKDD (2000)
16. Zhao, Y., Karypis, G.: Comparison of agglomerative and partitional document clustering algorithms. Technical Report 02-014, University of Minnesota (2002)
17. Zhong, S., Ghosh, J.: A comparative study of generative models for document clustering. In: Proc. of SDW Workshop on Clustering High Dimensional Data and its Applications, San Francisco, CA (2003)

An Effective Feature Selection Method for Text Categorization

Xipeng Qiu, Jinlong Zhou, and Xuanjing Huang

School of Computer Science, Fudan University, China
{xpqiu,10210240083,xjhuang}@fudan.edu.cn

Abstract. Feature selection is an efficient strategy to reduce the dimensionality of data and removing the noise in text categorization. However, most feature selection methods aim to remove non-informative features based on corpus statistics, which do not relate to the classification accuracy directly. In this paper, we propose an effective feature selection method, which aims at the classification accuracy of KNN. Our experiments show that our method is better than the traditional methods, and it is also beneficial to other classifiers, such as Support Vector Machines (SVM).

1 Introduction

Text categorization [17] is the problem of automatically classifying the natural language text to predefined categories based on their content. Text categorization plays an importance role in many applications, such as information retrieval [1] and Anti-Spam Email Filtering [5], etc.

In recent years, research interest in text categorization has been growing in machine learning, as well as in information retrieval, computational linguistics, and other fields. Statistical machine learning methods have shown great benefit over rule-based approaches in text categorization [20]. The top-performing learning methods include Naive Bayes [14], KNN [20], Support Vector Machines(SVM) [10] and maximum entropy methods(ME) [15]. However, when these traditional machine learning methods are applied to large-scale text categorization, a major characteristic, or difficulty, is the high dimensionality of the feature space.

Amongst the above machine learning methods, KNN is the most sensitive to the dimensionality of the data because KNN suffers from two limitations. The first limitation is that KNN requires to keep all training instances and finds the nearest neighbors by searching the whole set of training instances. Therefore, the efficiency of KNN is very low when both the size and dimensionality of the training data are huge. The second limitation is that KNN is very sensitive to noise [6]. These two limitations show that KNN is low-efficient for large-scale and high-dimensional text data. Therefore, it is highly desired to reduce the dimensionality of the data by selecting the useful features and removing the noise.

The approaches for this purpose, dimensionality reduction, have attracted much attention recently since it make the text categorization task more efficient and save more storage space [21,13,11,8].

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 50–61, 2011.

In the text domain, the popular feature extraction algorithms include Document Frequency (DF), χ^2 statistics (CHI), Information Gain (IG), Mutual Information (MI) , etc [21] and Orthogonal Centroid Feature Selection(OCFS) [18].

The DF is to select the features with the first p largest document frequency. The basic assumption is that rare terms are either non-informative for categorization nor influential in global performance. The χ^2 statistics measures the lack of independence between t and c and can be compared to the χ^2 distribution with one degree of freedom to judge extremeness. The orthogonal centroid feature selection (OCFS) [18] selects features optimally according to the objective function implied by the Orthogonal Centroid algorithm [9].

These methods aim to remove noninformative features according to corpus statistics. The criterion used in these methods did not relate to the classification accuracy directly [3]. Besides, they did not consider the particularity of text categorization.

In this paper, we propose an optimal feature selection method, called KNNFS, in text categorization. KNNFS has two major characteristics. One is that cosine similarity is taken in our feature selection criterion since it is usually used as similarity measure in text categorization; another is that our method adopts a criterion from the viewpoint of KNN. We calculate a score , called KNN score, for each feature and select the features with higher scores. Our experimental results show that our approach is better than the traditional FS methods, and it is also beneficial to other classifiers, such as Support Vector Machines (SVM).

The rest of the paper is organized as follows: Section 2 gives the review and analysis of the traditional feature selection methods in text categorization. Then we describe our optimal feature selection method, KNNFS, in Section 3. Experimental evaluations of our method and the other feature selection methods are presented in Section 4. Finally, we give the conclusions in Section 5.

2 Reviews of Feature Selection Methods in Text Categorization

In this section, we review the representative feature selection methods, such as document frequency [21], χ^2 statistics [21] and orthogonal centroid feature selection [18], which are used as baseline methods in our experiments.

2.1 Definitions

Firstly, we give some definitions used in this paper.

In text categorization, a corpus of documents is often mathematically represented by a $d \times n$ document matrix $X \in R^{d \times n}$, which is generated by the traditional TF·IDF indexing in Vector Space Model (VSM) [16], where n is the number of documents, and d is the number of features (terms). Each document is denoted by a column vector $x_i, (i = 1, 2, \cdots, n)$. The k^{th} entry, or feature, of x_i is denoted by $x_i^k, (k = 1, 2, \cdots, d)$. X^T is used to denote the transpose of X. The set of predefined categories is $\Omega = [\omega_1, \cdots, \omega_C]$, where C is the number of the categories.

We formulate the feature selection problem as follow: given a collection of n documents $X = \{x_1, x_2, \cdots, x_n\}$ with labels $Y = \{y_1, y_2, \cdots, y_n\}$ and $y_i \in \Omega, (i =$

$1, \cdots, n$), the task of feature selection is to find a subset of features indexed by s_l, ($l = 1, 2, \cdots, p$) such that the low dimensional representation of original data x_i is denoted by $\hat{x} = \{x_i^{s_1}, x_i^{s_2}, \cdots, x_i^{s_p})\}$.

2.2 Three Baseline FS Methods

Document Frequency(DF) Document frequency (DF) is the simplest feature selection technique for text categorization.

Document frequency is the number of documents in which a term occurs. We calculate the document frequency for each unique term in the training corpus. Then we select the features with the first p largest DF or remove features whose DF is less than a predefined threshold. The basic assumption is that rare terms are either non-informative for categorization nor influential in global performance.

χ^2 statistics(CHI). CHI is aiming at maximizing a criterion $J(W)$. It is also a greedy algorithm to save the computation cost and thus is not optimal either. To a given term t and a category ω_c , suppose A is the number of times t and ω_c co-occur, B is the number of times the t occurs without ω_c , C is the number of times ω_c occurs without t, D is the number of times neither ω_c nor t occurs. The χ^2 statistics is:

$$\chi^2(t, \omega_c) = \frac{n(AD - CB)^2}{(A + C)(B + D)(A + B)(C + D)}. \tag{1}$$

We can compute the χ^2 statistics between each unique term and each category in a training corpus, and then combine the category specific scores of each term into two scores:

$$\chi^2_{avg}(t) = \sum_{c=1}^{C} Pr(\omega_c)\chi^2(t, \omega_c), \tag{2}$$

$$\chi^2_{max}(t) = \max_{c=1}^{C} \chi^2(t, \omega_c). \tag{3}$$

The χ^2 statistics is known not to be reliable for low-frequency terms [7].

In this paper, we choose χ^2_{max} as one of our feature selection baseline methods.

Orthogonal Centroid Feature Selection(OCFS). Document frequency is the number of documents in which a term occurs. We calculate the document frequency for each unique term in the training corpus. Then we select the features with the first p largest DF or remove features whose DF is less than a predefined threshold. The basic assumption is that rare terms are either non-informative for categorization nor influential in global performance. The χ^2 statistics measures the lack of independence between t and c and can be compared to the χ^2 distribution with one degree of freedom to judge extremeness.

The orthogonal centroid feature selection (OCFS) [18] selects features optimally according to the objective function implied by the Orthogonal Centroid algorithm [9]. Orthogonal Centroid is a feature extraction method with projecting the feature space

into the orthogonal subspace spanned by all the centroids of the categories. OCFS optimizes the objective function of Orthogonal Centroid subspace learning algorithm in a discrete solution space.

OCFS computes the centroid for each category firstly. Using ω_c to represent class $c, (c = 1, \cdots, C)$, the mean vector of the c-th category is

$$m_c = \frac{1}{n_c} \sum_{x_i \in \omega_c} x_i \tag{4}$$

The mean vector of all documents is

$$m = \frac{1}{n} \sum_{i=1}^{n} x_i = \frac{1}{n} \sum_{c=1}^{C} n_c m_c \tag{5}$$

Then OCFS finds the p largest features from $\sum_{c=1}^{C} \frac{n_c}{c}(m_c^i - m^i)^2, (k = 1, \cdots, d)$.

Figure 1 gives the flowchart of OCFS.

1. compute the centroid $m_c, i = 1, 2, \cdots, C$ of each class for training data;
2. compute the centroid m of all training samples;
3. compute feature score $s(i) = \sum_{c=1}^{C} \frac{n_c}{c}(m_c^i - m^i)^2$ for all the features;
4. find the corresponding index set K consisted of the p largest ones in set $S = s(i)|1 \leq i \leq d$.

Fig. 1. Orthogonal Centroid Feature Selection Algorithm

2.3 Analysis of the Traditional Feature Selection Methods in Text Categorization

Although there are many other feature selection methods used in text categorization, such as information gain (IG) and mutual information (IM). It is shown that CHI, DF and OCFS give the competitive performance with IG and MI [21, 18].

Since OCFS is derived from the Orthogonal Centroid algorithm, it need the strong assumption that all categories are separated well by the centroids of the classes. Therefore, OCFS fails when the centroids of the classes coincide.

3 Optimal Feature Selection for KNN

In this section, we give the KNN decision rule for text categorization firstly, then we derive our feature selection criterion from the KNN decision rule.

3.1 K-Nearest Neighbor Classification(KNN) for Text Categorization

The KNN algorithm is quite simple: given a test document, the system finds the k nearest neighbors among the training documents, and uses the categories of the k neighbors

to weight the category candidates. The similarity score of each neighbor document to the test document is used as the weight of the categories of the neighbor document. If several of the k nearest neighbors share a category, then the per-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. By sorting the scores of candidate categories, a ranked list is obtained for the test document. By thresholding on these scores, binary category assignments are obtained. The decision rule in KNN can be written as:

$$y(x,\omega_c) = sign\left(\sum_{x_i \in KNN(x)} sim(x,x_i)y(x_i,\omega_c) - b_c\right) \tag{6}$$

where $y(x_i,\omega_c) \in \{0,1\}$ is the indicator for document x_i with respect to category ω_c; KNN(x) is the set of the k nearest neighbors of x; $sim(x,x_i)$ is the similarity between the test document x and the training document x_i and b_c is the category-specific threshold for the binary decisions.

Usually, the cosine value of two vectors is used to measure the similarity of the two documents. Thus, the KNN similarity of the document x and category ω_c is

$$sim(x,\omega_c) = \sum_{x_i \in KNN(x)} \frac{x^T x_i}{\sqrt{||x||}\sqrt{||x_i||}} y(x_i,\omega_c), \tag{7}$$

where $||x||$ is norm of x ($||x|| = \sqrt{x^T x}$).

3.2 Effective Feature Selection Criterion for KNN

To improve the performance of KNN, we should select the features so that the documents have higher KNN similarities (Eq.(7))with their corresponding categories and have lower KNN similarities with the other categories.

Given the training dataset $X = \{x_i, y_i\}, (i = 1, \cdots, n)$, where x_i is the i-th document and y_i is the corresponding label, we can calculate the KNN similarity of x_i and its corresponding category as

$$sim^{+}_{KNN}(x_i) = \sum_{\substack{x_j \in KNN(x) \\ y_j = y_i}} \frac{x_i^T x_j}{\sqrt{||x_i||}\sqrt{||x_j||}}, \tag{8}$$

and the KNN similarity of x_i and the other categories is calculated as

$$sim^{-}_{KNN}(x_i) = \sum_{\substack{x_j \in KNN(x) \\ y_j \neq y_i}} \frac{x_i^T x_j}{\sqrt{||x_i||}\sqrt{||x_j||}}. \tag{9}$$

From the viewpoint of KNN, we should select the features which have importance contributions in KNN similarity measure so that $sim^{+}_{KNN}(x_i)$ is as large as possible and $sim^{-}_{KNN}(x_i)$ is as small as possible for each document x_i.

We define the **KNN margin** of x_i as

$$Margin_{KNN}(x_i) = sim^{+}_{KNN}(x_i) - sim^{-}_{KNN}(x_i). \tag{10}$$

So we wish to select the features to make the KNN margin as large as possible. The larger the margin of between $sim^{+}_{KNN}(x_i)$ and $sim^{-}_{KNN}(x_i)$ is, the more accurate the results of categorization are.

Thus, the total KNN margin in training set can be written as

$$\begin{aligned} J &= \sum_i Margin_{KNN}(x_i) \\ &= \sum_i \left(\sum_{\substack{x_j \in KNN(x) \\ y_j = y_i}} \frac{x_i^T x_j}{\sqrt{||x_i||}\sqrt{||x_j||}} - \sum_{\substack{x_j \in KNN(x) \\ y_j \neq y_i}} \frac{x_i^T x_j}{\sqrt{||x_i||}\sqrt{||x_j||}} \right) \\ &= \sum_{i,j} x_i^T x_j W_{ij}, \end{aligned} \tag{11}$$

where W is a sparse symmetric $n \times n$ matrix with W_{ij}.

$$W_{ij} = \begin{cases} \frac{1}{\sqrt{||x_i||}\sqrt{||x_j||}} & : \quad j \in KNN(i), y_i = y_j \\ \frac{-1}{\sqrt{||x_i||}\sqrt{||x_j||}} & : \quad j \in KNN(i), y_i \neq y_j \\ 0 & : \quad j \notin KNN(i) \end{cases} \tag{12}$$

By simple algebra formulation, the total KNN margin can be written as:

$$\begin{aligned} J &= \sum_{i,j} x_i^T x_j W_{ij} = \sum_{i,j} \left(\sum_k x_i^k x_j^k \right) W_{ij} \\ &= \sum_k \left(\sum_{i,j} x_i^k W_{ij} x_j^k \right) \\ &= \sum_k \left(X^k W (X^k)^T \right) \\ &= \sum_k F(k) \end{aligned} \tag{13}$$

where $F(k) = X^k W (X^k)^T$.

For the k-th feature, the larger $F(k)$ is, the more its contribution is to maximize the total KNN margin. We call $F(k)$ the *KNN score* and use it as our criterion of feature selection.

Alg. 1 gives the flowchart of KNNFS.

1. compute the matrix W with Eq.(12);
2. compute the KNN score for each feature: $F(k) = X^k W (X^k)^T$;
3. find the corresponding index set K consisting of the p largest ones in set $S = \{s(i)|1 \leq i \leq d\}$.

Alg. 1. The proposed KNNFS Algorithm

3.3 An Illustrating Example

To give an intuitive comparison between the OCFS [18] and KNNFS, we apply them to select two features for 2D visualization on the 3D artificial dataset. The dataset is constituted by three classes. One class is generated from single Gaussian distribution, and both the rest two classes are generated with the mixture of two Gaussian distributions. The centroids of three classes coincide. Figure 2 demonstrates the results of 2D visualization by OCFS and KNNFS. KNNFS gives the perfect result, but OCFS fails since all the feature scores are zeros.

4 Experiments

In this section, we conduct our experiments on three real large scale text data sets to show the performance of KNNFS. We first describe the experiments setup, then give the experimental results, and finally discuss the results.

4.1 Datasets

To demonstrate the efficacy of KNNFS, we performed experiments on three data sets: Reuters21587 [14], 20 Newsgroups [12] and WebKB [4].

Reuters21587. The ApteMod version of Reuters21587 [14] which was obtained by eliminating unlabeled documents and selecting the categories which have at least one document in the training set and the test set. This process resulted in 90 categories in both the training and test sets. After eliminating documents which do not belong to any of these 90 categories, we obtained a training set of 7768 documents, a test set of 3019 documents, and a vocabulary of 23793 unique words after stemming and stop word removal. The number of categories per document is 1.23 on average. The category distribution is skewed; the most common category has a training set frequency of 2877 but 82% of the categories have less than 100 instances, and 33% of the categories have less than 10 instances.

20 Newsgroups. The 20 Newsgroups data consists of Usenet articles Lang collected from 20 different newsgroups [12]. Over a period of time 1000 articles were taken from each of the newsgroups, which make up of an overall number of 20000 documents in this collection. Except for a small fraction of the articles, each document belongs to

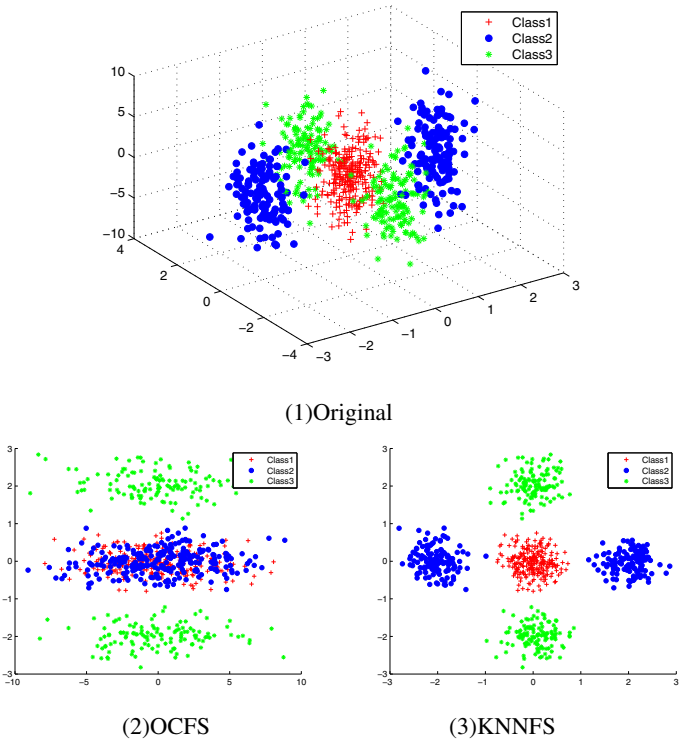


Fig. 2. 2D visualization of OCFS and KNNFS on 3D artificial dataset

exactly one newsgroup. We use the "bydate" version of data whose training and testing data are split previously by the data provider[1]. There are 18941 documents and 70776 unique words after stemming and stop word removal.

WebKB. The WebKB data set [4] contains web pages gathered from university computer science departments. The pages are divided into seven categories: student, faculty, staff, course, project, department and other. In this paper, we use the four most popular entity-representing categories: student, faculty, course and project, all together containing 4199 pages. We just remove the html tags ('</p>', '</h1>', etc.) and stop words. After stemming, the resulting vocabulary has 22021 words. We use four-fold cross validation by training on three of the universities plus the misc collection, and testing on the pages from a fourth, held-out university.

4.2 Classifiers

In order to compare the different feature selection methods, we apply them to the three categorization method: KNN with RCut [19], KNN with SCut [21,19] and SVM [2].

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/

KNN with RCut. The RCut is a thresholding strategies [19]. In RCut, we sort categories by score for each document, and assign this document to each of t top-ranking categories. RCut is parameterized by t which can be either specified manually or automatically tuned using a validation set. In this paper, we set $t = 1$ since that the number of categories per document is closer to 1 in our experiments.

KNN with SCut. The formulation of KNN with SCut is in Eq.(6). We tune the threshold b_c for each category by a validation set of documents. The category-specific threshold b_c is automatically learned using a validation set of documents. That is, we used a subset of the training documents not used the test documents to learn the optimal threshold for each category. By optimization, we mean the threshold that yield the best F_1 score on the validation documents.

In this paper, we use leave-one-out cross validation to choose the optimal thresholds.

SVM. Besides KNN classification, we also test the performance of KNNFS with SVM to see whether KNNFS is helpful for the other classifiers. For SVM classifier, we choose SVMlib [2] with multi-class classification. We use the radial basis kernel in our experiments.

4.3 Performance Measurement

In this section, we use micro-averaging $F1$ [20] to measure the performance of each result.

4.4 Experimental Results

Reuters21587. The performances of the different feature selection algorithms on Reuters21587 data are reported in Table 1. From this table, we can see that the low dimensional spaces selected by KNNFS have better performance than its counterpart selected by the other methods(DF, CHI and OCFS). CHI gives poor results in the lower dimension, but becomes better with the increasing of the dimensionality. It is also shown that all methods are overfit when the dimensionality is over 1024 because the noise or redundant features could appear. Besides, the performance of SVM with KNNFS is better than the other FS methods, which indicates that KNNFS not only improves the performance of KNN, but is helpful for SVM.

20 Newsgroups. The performances of the different feature selection algorithms on 20 Newsgroups data are reported in Table 2. It is also shown that KNNFS has better performance than the other FS methods. There is not overfit for each method. The reason may be that the original feature dimensionality is very large and the selected features are just little proportion of the original features. Therefore, there may have the fewer redundant features in selected features.

WebKB. The performances of the different feature selection algorithms on WebKB data are reported in Table 3. It is also proved that KNNFS outperform the other FS methods. Besides, all methods are overfit in high dimensional space.

Table 1. Micro-averaging $F1$ on Reuters21587 dataset

Classifier	FS	Number of Features												
		2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
RCut KNN	DF	21.3	38.2	40.4	52.9	56.4	62.0	67.3	70.4	73.0	73.9	76.4	75.6	75.8
	CHI	21.3	21.3	21.4	26.7	30.2	31.1	64.9	73.7	76.4	79.6	79.3	76.1	75.1
	OCFS	23.8	24.3	26.2	41.0	51.6	68.9	73.0	72.8	76.1	76.7	75.9	75.3	75.0
	KNN	**27.2**	**40.4**	**48.8**	**55.0**	**62.6**	**69.4**	**73.4**	**76.9**	**78.8**	**80.6**	**80.2**	**80.8**	**80.4**
SCut KNN	DF	**13.6**	19.0	28.5	35.8	41.2	48.0	56.9	62.1	69.8	73.7	77.4	77.8	77.6
	CHI	0	0.1	0.4	14.0	32.5	37.4	53.4	64.1	70.4	76.3	78.5	77.5	77.7
	OCFS	9.4	15.5	20.1	35.2	41.7	56.6	66.2	68.8	**73.9**	76.5	77.7	78.1	77.7
	KNN	8.4	**21.3**	**35.9**	**40.8**	**44.9**	**57.8**	**68.6**	**70.2**	72.7	**78.4**	**80.8**	**81.3**	**81.3**
SVM	DF	**38.4**	38.2	43.2	53.7	59.3	64.8	70.4	72.4	73.9	74.2	74.0	72.0	70.4
	CHI	32.1	32.1	32.2	37.6	55.3	55.9	67.8	74.1	77.5	78.9	77.2	74.1	70.5
	OCFS	34.9	40.0	44.1	53.5	58.6	71.6	76.1	77.1	77.6	76.6	74.2	71.7	70.3
	KNN	32.1	**40.1**	**51.3**	**57.0**	**62.5**	**73.7**	**76.3**	**76.7**	**77.8**	**79.0**	**78.3**	**75.0**	**70.9**

Table 2. Micro-averaging $F1$ on 20 Newsgroups dataset

Classifier	FS	Number of Features												
		2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
RCut KNN	DF	4.9	6.3	9.8	12.6	16.1	20.7	29.6	40.1	50.8	58.4	64.4	68.7	72.5
	CHI	**8.6**	11.5	20.8	27.3	32.9	45.9	54.5	59.8	65.3	69.6	71.9	73.9	74.5
	OCFS	4.3	4.7	5.0	7.5	12.4	29.4	46.6	56.1	61.0	64.5	67.9	71.3	73.4
	KNN	8.4	**13.7**	**25.7**	**31.2**	**36.8**	**46.7**	**55.5**	**61.9**	**66.3**	**70.9**	**72.9**	**75.0**	**76.2**
SCut KNN	DF	**9.4**	9.7	12.1	13.8	17.0	21.0	30.2	39.8	49.5	55.2	61.0	65.7	68.7
	CHI	7.8	11.7	20.6	24.4	33.2	41.6	47.3	50.8	54.9	59.6	65.6	69.6	71.8
	OCFS	1.7	3.7	7.5	10.6	13.3	22.3	35.4	45.4	53.0	59.3	64.7	68.4	69.9
	KNN	7.8	**13.7**	**21.0**	**27.7**	**35.7**	**46.2**	**47.8**	**53.2**	**55.1**	**61.1**	**66.0**	**70.6**	**73.2**
SVM	DF	5.9	7.2	12.1	13.6	19.5	27.2	39.3	49.7	62.3	69.6	75.0	77.4	78.7
	CHI	**9.6**	12.6	22.0	28.3	41.8	51.1	60.2	65.0	70.5	74.6	77.9	79.7	80.7
	OCFS	5.4	5.9	9.4	10.2	15.4	33.2	52.0	61.4	68.3	73.0	76.8	78.9	79.7
	KNN	8.4	**13.9**	**23.1**	**29.1**	**45.8**	**54.9**	**63.7**	**65.2**	**71.8**	**76.5**	**78.6**	**79.3**	**81.2**

4.5 Discussion of Results

From the experiments we can see that the proposed KNNFS is better than DF, CHI and OCFS in most cases. From the experimental results, we can draw the conclusion that KNNFS can get significant improvements than baselines, since it is an effective feature selection approach from the KNN decision rule directly, can outperform the greedy ones.

Besides, we notice that the accuracies of KNNFS are lower than other methods when the number of selected features is 2. This phenomenon occurs due to the overfitting when the selected feature dimension is small.

There are two reasons why KNNFS outperforms the other methods. One is that KNNFS selects features from the viewpoint of KNN directly. Another is that KNNFS maximizes the KNN margin, which can reduce the generalization error potentially.

Table 3. Micro-averaging $F1$ on WebKB dataset

Classifier	FS	Number of Features										
		2	4	8	16	32	64	128	256	512	1024	2048
RCut KNN	DF	23.0	26.2	19.5	58.3	66.7	70.2	72.8	75.7	74.9	75.3	75.8
	CHI	**23.4**	**33.7**	35.4	50.2	74.2	79.5	81.5	81.7	80.0	79.0	**76.8**
	OCFS	**23.4**	24.9	24.9	36.3	63.7	69.5	71.1	71.8	74.7	75.4	76.2
	KNN	**23.4**	33.0	**36.9**	**55.2**	**79.2**	**80.0**	**83.5**	**83.9**	**81.3**	**80.3**	76.6
SCut KNN	DF	24.1	26.9	29.8	41.6	51.3	53.0	56.0	61.9	66.7	70.7	74.2
	CHI	**29.7**	38.3	39.5	45.2	51.7	58.4	64.0	65.9	68.8	72.8	74.9
	OCFS	26.8	29.3	30.5	41.5	50.5	54.7	55.8	58.7	64.8	70.1	74.2
	KNN	26.8	**39.9**	**43.3**	**45.9**	**55.9**	**60.7**	**64.5**	**68.8**	**69.2**	**73.4**	**75.4**
SVM	DF	64.3	60.9	59.6	61.6	68.0	76.3	81.9	84.5	84.8	83.5	83.3
	CHI	**73.1**	78.4	79.3	**79.7**	76.0	83.4	84.9	86.1	86.4	85.2	84.5
	OCFS	68.8	70.2	71.1	65.7	70.1	76.8	79.3	82.8	83.3	82.4	82.3
	KNN	68.8	**79.5**	**79.4**	**79.7**	**81.1**	**83.2**	**85.4**	**89.8**	**89.2**	**85.7**	**85.3**

Similar with the other methods, the accuracies of KNNFS also drop when the number of selected features is enough large. The reason is that there is no measure of redundancy between selected features, which could destroy the performance of KNNFS.

5 Conclusion

In this paper, we propose a new feature selection method, called KNNFS, for text categorization. There are two contributions in our work. One is that our method adopts a criterion from the viewpoint of KNN, which selects useful features according to the KNN decision rule in text categorization directly. Another is that cosine similarity, instead of Euclidean distance, is adopted since it is usually used as similarity measure in text categorization.

In the future, we will investigate how to learn the best number of selected features automatically based on KNN margin. In addition, we will also extend our feature selection criterion by imposing the penalty to the redundant features.

Acknowledgments

This work was (partially) funded by NSFC (No. 61003091), Shanghai Leading Academic Discipline Project (No. B114), Shanghai Committee of Science and Technology (No. 10dz1500102) and 863 Program (No. 2009AA01A346).

References

1. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval. Addison-Wesley, Harlow (1999)
2. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), http://www.csie.ntu.edu.tw/~cjlin/libsvm

3. Chen, X., Wasikowski, M.: Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 124–132. ACM, New York (2008)
4. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S.: Learning to extract symbolic knowledge from the world wide web. In: Proc. of National Conf. on Artif. Intell. (AAAI), pp. 509–516 (1998)
5. Drucker, H., Wu, D., Vapnik, V.: Support vector machines for spam categorization. IEEE Transactions on Neural Networks 10(5), 1048–1054 (1999)
6. Duda, R., Hart, P., Stork, D.: Pattern Classification, 2nd edn. Wiley, New York (2001)
7. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. Computational Linguistics 19(1), 61–74 (1993)
8. Galavotti, L., Sebastiani, F., Simi, M.: Experiments on the use of feature selection and negative evidence in automated text categorization. In: Research and Advanced Technology for Digital Libraries, pp. 59–68 (2010)
9. Howland, P., Park, H.: Generalizing discriminant analysis using the generalized singular value decomposition. IEEE Trans. Pattern Anal. Machine Intell. 26(8), 995–1006 (2004)
10. Joachims, T.: Text categorization with suport vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
11. Kim, H., Howland, P., Park, H.: Dimension reduction in text classification with support vector machines. J. Mach. Learn. Res. 6, 37–53 (2005)
12. Lang, K.: Newsweeder: Learning to filter netnews. In: Proc. of Int. Conf. on Mach. Learn. (ICML), pp. 331–339 (1995)
13. Lee, C., Ha, J., Lee, G.G.: Mmr-based feature selection for text categorization. In: Proceedings of the HLT/NAACL 2004 (2004)
14. Lewis, D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Third Annual Symposium on Document Analysis and Information Retrieval, pp. 81–93 (1994)
15. Nigam, K., Lafferty, J., McCallum, A.: Using maximum entropy for text classification. In: IJCAI 1999 Workshop on Machine Learning for Information Filtering, pp. 61–67 (1999)
16. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
17. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
18. Yan, J., Liu, N., Zhang, B., Yan, S., Chen, Z., Cheng, Q., Fan, W., Ma, W.: Ocfs: optimal orthogonal centroid feature selection for text categorization. In: Proc. of Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR), pp. 122–129. ACM Press, New York (2005)
19. Yang, Y.: A study of thresholding strategies for text categorization. In: Proc. of Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR), pp. 137–145. ACM Press, New York (2001)
20. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proc. of SIGIR. ACM Press, New York (1999)
21. Yang, Y., Pedersen, J.: A comparative study on feature selection in text categorization. In: Proc. of Int. Conf. on Mach. Learn. (ICML), vol. 97 (1997)

A Subpath Kernel for Rooted Unordered Trees

Daisuke Kimura[1], Tetsuji Kuboyama[2], Tetsuo Shibuya[3], and Hisashi Kashima[1]

[1] Graduate School of Information Science and Technology, The University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo, 113-8656, Japan
Daisuke_Kimura@mist.i.u-tokyo.ac.jp, kashima@mist.i.u-tokyo.ac.jp
[2] Computer Centre, Gakushuin University, Mejiro 1-5-1, Toyoshima-ku, Tokyo, 171-8588, Japan
kuboyama@tk.cc.gakushuin.ac.jp
[3] Human Genome Center, Institute of Medical Science, The University of Tokyo, Shirokanedai 4-6-1, Minato-ku, Tokyo, 108-8639, Japan
tshibuya@hgc.jp

Abstract. Kernel method is one of the promising approaches to learning with tree-structured data, and various efficient tree kernels have been proposed to capture informative structures in trees. In this paper, we propose a new tree kernel function based on "subpath sets" to capture vertical structures in rooted unordered trees, since such tree-structures are often used to code hierarchical information in data. We also propose a simple and efficient algorithm for computing the kernel by extending the multikey quicksort algorithm used for sorting strings. The time complexity of the algorithm is $O((|T_1|+|T_2|)\log(|T_1|+|T_2|))$ time on average, and the space complexity is $O(|T_1|+|T_2|)$, where $|T_1|$ and $|T_2|$ are the numbers of nodes in two trees T_1 and T_2. We apply the proposed kernel to two supervised classification tasks, XML classification in web mining and glycan classification in bioinformatics. The experimental results show that the predictive performance of the proposed kernel is competitive with that of the existing efficient tree kernel for unordered trees proposed by Vishwanathan *et al.* [1], and is also empirically faster than the existing kernel.

Keywords: Kernel methods, tree kernels, convolution kernels.

1 Introduction

Recently, machine learning methods for complex-structured data such as strings, trees and graphs, have been becoming increasingly important. In natural language processing, sentences are often represented as strings, and parsed sentences are represented as trees [2]. In bioinformatics, there are various kinds of sequential-structure data such as DNAs, RNAs and proteins, and tree-structured data such as glycans and secondary structures of RNAs. Sometimes structures of chemical compounds and proteins are represented as graph-structured data. In Web mining, documents are written in tree-structured XML and HTML, and access and purchase behaviors of visitors to Web pages are represented as trees

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 62–74, 2011.

or graphs. Effective and efficient analysis of such complex-structured data play key roles in businesses, healthcare and scientific researches.

In this paper, we focus on learning with tree-structured data, more specifically, rooted unordered trees. In general learning problems, data are represented as vectors in a feature space. If we can properly define the bases of the feature space, we can just pass the feature vectors to learning algorithms. However, when we handle more complex-structured data such as sequences, trees, and graphs that have structures among their constituent elements, proper vector representation is not obvious, and the step of feature design can be difficult and time-consuming.

Probably, one of the reasonable strategies for handling such complex-structured data is to use their local structures as features, for example, substrings and subsequences for strings, subtrees and tree-structured subgraphs for trees, and paths and subgraphs for graphs. The feature vectors can be constructed by using the numbers of times such substructures appear in target data as features. However, as the numbers of such substructures are enormous, it is sometimes prohibitive to explicitly enumerate them to construct feature vectors. There is also another serious problem called "curse of dimensionality", which is degradation of predictive performance in high-dimensional feature spaces.

Kernel methods [3] are one of the promising approaches to learning with complex-structured data. Kernel methods use only the inner products of feature vectors when they access data. This means that even if the dimensionality of the feature vector is extremely large, the dimensionality does not explicitly affect the computational cost as long as efficient procedures to compute the inner products are available. The functions computing the inner products efficiently are called "kernel functions", and kernel methods can work efficiently in high dimensional feature spaces by using kernel functions. Moreover, a well-known kernel method, the support vector machine(SVM), is known to have good generalization properties, both theoretically and experimentally, and overcome the "curse of dimensionality" problem in high dimensional feature spaces [4].

Haussler [5] introduced convolution kernels, a general framework for handling discrete data structures with kernel methods. In convolution kernels, structured data are decomposed into their parts, and kernel functions are defined in terms of the parts. By using this framework, Collins and Duffy [6] designed a convolution kernel for parse trees used in natural language processing. They implicitly constructed feature vectors by using the number of times each tree-structured subgraph appears in parse trees, and defined a kernel function as the inner product of those feature vectors. They proposed an efficient kernel computation algorithm by using dynamic programming. After the seminal work by Collins and Duffy [6], various kernels for more general trees were developed including labeled ordered trees [7,8] and variants of ordered trees such as positional trees [9] and syntactic trees [10]. However, there are only a few works on kernel design for rooted unordered trees because of the hardness result for computing general kernels for unordered trees [11]. Vishwanathan *et al.* [1] proposed a simple but efficient kernel for unordered trees based on subtrees. It converts trees into

strings, and boils down tree comparison to string comparison. By employing appropriate data structures such as suffix trees and suffix arrays, their kernel can be computed in linear time, that is, $O(|T_1| + |T_2|)$. The feature space spanned by the linear-time tree kernel is constructed not by using tree-structured subgraphs, but by using subtrees. Therefore, the kernel is suitable for capturing horizontal substructures, but not for vertical substructures such as paths from root to leaves.

On the other hand, in the context of information retrieval, Ichikawa *et al.* [12] proposed a tree similarity using subpaths of paths from tree root to leaves, and showed its effectiveness in text retrieval tasks. Inspired by their work, we propose a new tree kernel for rooted unordered trees based on tree subpaths, and a fast and space-efficient algorithm to compute the kernel by extending the multikey quicksort algorithm [13] for sorting strings. The computational complexity of our kernel is $O((|T_1| + |T_2|) \log(|T_1| + |T_2|))$ on average, which is worse than the linear-time tree kernel by vishwanathan *et al.* [1]. However, in practice, it is very efficient, and actually, faster than the linear-time kernel in our experiments.

Finally, we demonstrate the predictive accuracy and efficiency of the proposed kernel by using two tasks, XML classification task in Web mining, and glycan classification tasks in bioinformatics. The results show our kernel is competitive with the linear-time tree kernel in predictive performance, and is faster than the linear-time kernel.

This paper is organized as follows. Section 2 reviews the existing tree kernels such as the parse tree kernel [6], the labeled ordered tree kernel [7], and especially the linear-time tree kernel [1]. In Section 3, based on the idea of tree subpaths [12], we propose a new tree kernel by using the subpaths. We also propose an efficient algorithm for computing the proposed kernel function. In Section 4, we show experimental results on two kinds of classification tasks, one with XML data, and the other one with glycan data. Section 5 reviews the related work, and Section 6 gives discussion and future work, and concludes this paper.

2 A Linear-Time Kernel for Rooted Unordered Trees

Vishwanathan *et al.* [1] proposed a linear-time kernel for rooted labeled unordered trees, whose computational complexity is $O(|T_1| + |T_2|)$. The substructures that the kernel can use as features are restricted to a subset of those used by the previous tree kernels. However, thanks to this simplification, they achieved the linear-time complexity.

The basic idea of their tree kernel is to convert two trees into two strings and then apply the string kernel to them. First, it converts the two trees into two string as follows. Let us denote by T one of the target trees, and by $label(n_s)$ the label of a node n_s in T. $tag(n_s)$ denotes the string representation of the subtree of T rooted at n_s. Therefore, $tag(n_{root})$ is the string representation of T, where n_{root} is the root of T. We recursively apply the following steps in a bottom-up manner to obtain $tag(n_{root})$.

– If the node n_s is a leaf, let $tag(n_s) = [label(n_s)]$.
– If the node n_s is not a leaf and has c children $n_1, n_2, \ldots, n_c$, sort $tag(n_1), tag(n_2), ..., tag(n_c)$ in lexical order to obtain $tag(1), tag(2), ..., tag(c)$, and let $tag(n_s) = [label(n_s)tag(1)tag(2)...tag(c)]$.

Figure 1 shows an example of the tree-to-string conversion. In the resultant string, a substring starting from ' [' and ending at ']' with the same number of ' ['s and ']'s corresponds to a subtree rooted at a particular node.

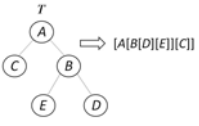


Fig. 1. An example of the tree-to-string conversion

We apply the above conversion to two trees T_1 and T_2 to obtain two strings S_1 and S_2. Now we define the tree kernel between T_1 and T_2 via the two strings S_1 and S_2 as

$$\begin{aligned} K(T_1, T_2) &= K(S_1, S_2) \\ &= \sum_{s \in \Sigma^*} w_{|s|} num(S_{1s}) num(S_{2s}), \end{aligned} \tag{1}$$

where Σ^* is the set of substrings of S_1 and S_2, $num(S_{1s})$ and $num(S_{2s})$ are the numbers of times a substring s appears in S_1 and S_2, respectively. $w_{|s|}$ is a weight parameter for substrings of length $|s|$. Typical choices for $w_{|s|}$ are:

– constant weighting (defined as $w_{|s|} = 1$),
– k-spectrum weighting (defined as $w_{|s|} = 1$ if $|s| = k$, and $w_{|s|} = 0$ otherwise), and
– exponential weighting (defined as $w_{|s|} = \lambda^{|s|}$ where $0 \leq \lambda < 1$ is a decaying rate).

To compute the kernel (1), it is sufficient to enumerate all of the common substrings of S_1 and S_2, and to count the numbers of times they occur in each of S_1 and S_2. It can be done in $O(|S_1| + |S_2|)$ time by employing suffix trees or suffix arrays [1,14]. If we assume the maximum number of bits needed to describe a node label is constant, the lengths of the converted strings are $|S_1| = O(|T_1|)$ and $|S_2| = O(|T_2|)$, respectively. Therefore, the computational complexity of computing the kernel function (1) is $O(|T_1| + |T_2|)$.

Despite the efficiency of the linear-time tree kernel, there is a drawback of the linear-time tree kernel. The drawback is that the kernel considers only subtrees as features, namely, subtrees were required to match some portion of the labeled unordered tree perfectly. Such a property is not preferred in the case of large trees and many kinds of node labels.

3 A New Tree Kernel Based on Tree Subpaths

3.1 Subpath Set

In this section, we propose another efficient tree kernel that can capture vertical structures in trees without false positive enumerations of common substructures. Since tree-structured data often represent hierarchical structures such as inclusive relations of tags in HTML and XML documents, it is important for tree kernels to take such vertical structures into account. We employ the idea of subpath sets introduced by Ichikawa *et al.* [12], who modeled a sentence as a tree and used tree similarities for information retrieval. A substring of a path from the root to one of the leaves is called a subpath, and a subpath set is a set of subpaths included in trees. They defined a similarity measure using the subpath set. Figure 2 shows an example of a subpath set.

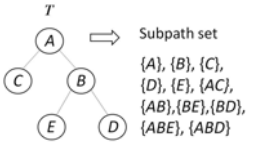


Fig. 2. An example of a subpath set

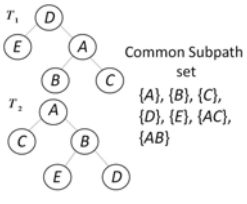


Fig. 3. An example of a common subpath set

3.2 A Subpath Tree Kernel

Let us assume that we want to define a tree kernel $K(T_1, T_2)$ between two trees T_1 and T_2. By using the subpath set,we define our tree kernel as

$$K(T_1, T_2) = \sum_{p \in S} w_{|p|} num(T_{1p}) num(T_{2p}), \tag{2}$$

where S is the subpath set of T_1 and T_2, $|p|$ is the number of nodes included in a subpath, and $num(T_{1p})$ and $num(T_{2p})$ are the numbers of a subpath p included in T_1 and T_2, respectively. $w_{|p|}$ is the weight parameter introduced in Section 2.

Therefore, the proposed kernel defines its features as a subpath set, and takes the inner product of two feature vectors with weights according to subpath lengths. Figure 3 shows an example of a common subpath set of two trees.

3.3 An Efficient Algorithm for the Subpath Tree Kernel

Since the size of a subpath set of a tree T is $O(|T|^2)$ generally, naive computation of the subpath tree kernel (2) costs $O(|T|^2)$-computation time. In this subsection, we propose a more efficient algorithm by extending the multikey quicksort algorithm [13].

The basic idea behind our algorithm is to enumerate prefixes of common suffixes of two trees, where a suffix of a tree is the string starting from a node and ending at the root. For example, in Figure 2, the set of suffixes of the tree is $\{A, BA, CA, DBA, EBA\}$. When the number of nodes in a tree is n, the number of suffixes is also n.

Let us denote a suffix starting from node n_1 by $label(n_1)label(n_2)...label(n_{root})$, where $label(m)$ denotes the label of node m and $label(n_{root})$ denotes the root label. Let d be the number of nodes included in this suffix. We call $label(n_1)label(n_2)...label(n_k)$ $(k \leq d)$ a prefix of suffix of n_1. Noting that a subpath set is defined as the substrings of a path from the root to a leaf, prefixes of all suffixes of a tree are identical to the set of (reversed) subpaths. Therefore, we can compute the kernel (2) by enumerating all prefixes of common suffixes of two trees.

Next, we describe an efficient way to enumerate all prefixes of common suffixes of two trees. We extend the idea of the multikey quicksort algorithm [13], which is a simple and efficient algorithm for sorting a set of strings. The multikey quicksort algorithm resembles the general quicksort algorithm, but is different from the quicksort in the following two points. (1) The multikey quicksort alphabetically compares labels in strings from their beginning to the end one by one (not the whole strings). The comparison starts at the first labels, and proceed to the next position if sorting is not completed yet. (2) At each step of the algorithm, the multikey quicksort sets a pivot, and divides the strings into three groups by comparing the labels at the current position with the pivot.

The following is a brief description of the algorithm of the multikey quicksort.

1. Initialize the current position h as $h := 1$, and the current set S as the set of all strings.
2. For strings included in the current set S,
   (a) Choose one of the labels at the current position as a pivot at random.
   (b) Compare the labels at the current position with the pivot label, and divide the strings into three sets S_{small}, S_{large} and S_{equal}, that are, strings with the current position labels alphabetically larger than the pivot, ones with smaller labels than the pivot, and the others (with the same labels with the pivot label), respectively.
   (c) Recursively apply Step 2 to each of S_{small} and S_{large} by setting $S := S_{small}$ or $S := S_{large}$.
   (d) Set $h := h + 1$.
   (e) Remove strings with no labels at h from S_{equal}, and apply Step 2 to S_{equal} by setting $S := S_{equal}$.

The multikey quicksort recursively divides the dataset by using pivots just as the quicksort algorithm with incrementing the current position. If the number of target strings is n, the multikey quicksort runs in $O(n\log n)$ on average, and $O(n^2)$ in the worst case.

Extending the idea of the multikey quicksort, the algorithm for computing the proposed kernel function (2) between two trees T_1 and T_2 is described as follows.

1. Initialize the kernel function value k as $k := 0$, the current position h as $h := 1$, and the current set S the set of all nodes in T_1 and T_2.
2. For nodes included in the current set S,
   (a) Choose one of the node labels in S as a pivot at random.
   (b) Compare the node labels in S with the pivot label, and divide the nodes into three sets S_{small}, S_{large} and S_{equal}, that are, nodes with labels alphabetically larger than the pivot, nodes with smaller labels than the pivot, and the others (with the same labels with the pivot label), respectively.
   (c) If S_{small} has at least one node from both of T_1 and T_2, apply Step 2 to $S := S_{small}$.
   (d) Apply Step 2(c) to S_{large}.
   (e) If S_{equal} has at least one node from both of T_1 and T_2, update k by using

$$k := k + w_h L(T_1) L(T_2)$$

   where w_h is the weight parameter, $L(T_1)$ and $L(T_2)$ are the number of nodes in S_{equal} originated from T_1 and T_2, respectively. Otherwise, exit from the current recursion. Set $h := h + 1$ and apply Step 2 to $S := parents(S_{equal})$, where $parents(S_{equal})$ denotes the set of parent nodes of S_{equal}.

The proposed algorithm recursively computes the kernel function in a bottom-up manner. At each recursion of the algorithm, we recursively divide a current node set into three non-overlapping sets just like the multikey quicksort. Therefore, the time complexity of the algorithm is $O((|T_1|+|T_2|)\log(|T_1|+|T_2|))$ on average and $O((|T_1| + |T_2|)^2)$ in the worst case, which is more efficient the original tree kernels. The time complexity is slightly worse than that of [1], but in practice, our algorithm is more efficient than the linear-time kernel, which will be demonstrated in the experimental section. Note that the pointers from nodes to parents can be constructed in linear time by using the depth first search.

Figure 4 shows an illustration of how the algorithm works for the two trees given in Figure 3. At the first step, the algorithm initializes the current position h and the kernel function value k as $h =: 1$ and $k := 0$, respectively. It sets the current set S as the union set of all nodes in T_1 and T_2. For example in Figure 3, $T_1 : D$ indicates that S includes a node with label 'D' in T_1. At Step 2(a), the algorithm chooses one of the node labels in S as a pivot at random. In this example, let us assume that we choose B as the pivot label. Then in Step 2(b), we compare the labels in S with the pivot label, and divide the nodes in the current set into three sets, $S_{small}(= \{T_1 : A, T_2 : A\})$, $S_{equal}(= \{T_1 : B, T_2 : B\})$, and $S_{large}(= \{T_1 : D, T_1 : E, T_1 : C, T_2 : C, T_2 : E, T_2 : D\})$. Finally, the algorithm recursively calls itself for each of S_{small} and S_{large} (at Step 2(c) and Step 2(d)). As for S_{equal}, it updates k and h, and recursively applies the algorithm for $S := parents(S_{equal})$ (at Step 2(e)).

This algorithm enumerates all of the prefixes of the common suffixes of two trees, but it does not explicitly construct all of the suffixes in memory. The memory required for the explicit construction is $O(|T_1|^2 + |T_2|^2)$, which can not achieve the time complexity of the proposed algorithm.

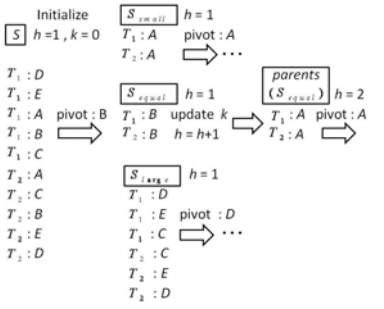


Fig. 4. A flowchart of our proposed algorithm applied to the two trees in Figure 3

We can imagine that other efficient data structures such as the suffix tree of a tree can be an option for efficient computation of the kernel. Several linear-time construction algorithms of the suffix tree of a tree were proposed, e.g. by [15]. However, to apply it to the kernel computation, if one node has several children with an identical label, we need to merge them to one node, which changes the structure of the target tree and results in imprecise kernel values.

In addition to the practical efficiency of the proposed algorithm, another virtue of the algorithm is its simpleness. This is contrast with the fact that efficient implementations of the linear-time approaches using suffix arrays or suffix trees are very complex.

4 Experiments

In this section, we demonstrate the performance of the proposed tree kernel for binary classification tasks by using three datasets.

First, we compare the predictive performance of the proposed kernel (denoted by 'Subpath") and the linear-time tree kernel [1] (denoted by 'Vishwanathan') combined with the three weighting schemes [14], the constant weighting (CW), the k-spectrum weighting (kSW) and the exponential weighting (EW) defined in Eqs. (2) and (1). We also use the three baseline kernels based on similarity measures proposed by Kailing *et al.* [16] (denoted by 'Kailing'). In Kailing kernels, each kernel is defined as the L_1 distance of histograms of heights, degrees or node labels. All of the performance results are evaluated in AUC measured by using the 10-fold cross-validation.

Next, we compare the execution time of the proposed kernel (Subpath) to the linear-time tree kernel (Vishwanathan). We use the the linear-time tree kernel implemented by Teo and Vishwanathan [17]. We run all the experiments on a Core2 Duo 2.00GHz Windows machine. In all of the experiments, we use LIBSVM [18] as the SVM implementation.

4.1 An XML Classification Dataset

In this subsection, we show experimental results using an XML dataset. We use the XML dataset provided by Zaki and Aggarwal [19]. The dataset collects

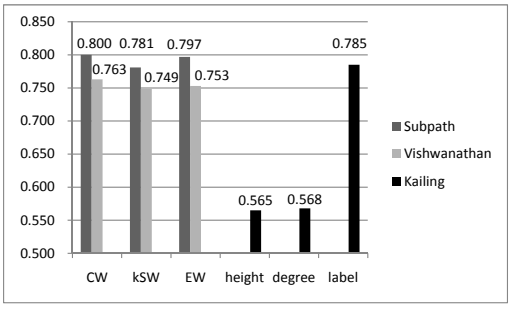


Fig. 5. Results for the XML classification task. The performances were measured in AUC with the 10-fold cross-validation.

the access behaviors of visitors within the Web site of some computer science department during three weeks. The access behaviors of a particular visitor are represented as an unordered tree. Each data is assigned either of two classes, 'edu' or 'other', where 'edu' means that the visitors came from educational domains such as universities, while 'other' corresponds to visitors from the other domains. The goal of this classification task is to discriminate visitors from 'edu' domain and those from the other domains. Since the dataset includes many small trees, we use only trees whose depths are more than 4. The size of the resultant dataset is 3,183 (773 'edu's and 2,410 'other's).

We show the SVM classification results with the XML dataset in Figure 5. For Subpath kernel and Vishwanathan with the three weighting schemes, the constant weighting (CW), the k-spectrum weighting (kSW) and the exponential weighting (EW)[2]. The hyper-parameters are determined by using cross-validation.

The AUCs of the Subpath kernels with two weighting schemes (CW and EW) are higher than the Kailing kernel using node labels. Since the Kailing kernel based on node labels does not consider tree structures at all, this result shows that Subpath kernel can capture tree structures appropriately. Moreover, the AUC of the Subpath kernel is the highest among all of the results. In contrast to Subpath kernel, the AUCs of Vishwanathan kernel are lower than those of the Kailing kernel. This is probably because the number of different node label is very large in this XML dataset, and therefore subtrees are not appropriate as features. According to the results, the proposed Subpath kernel performs well in the XML classification task.

4.2 Glycan Classification Datasets

Next, we show experimental results on glycan classification, which is an important task in bioinformatics. Glycans are carbohydrate chains, which are considered to play an important role in various fundamental biological processes such as cell-cell interaction. The structure of a glycan is abstractly represented as a

tree structure by representing carbohydrates as nodes and their covalent bonds as edges. The glycan structure dataset used in the experiment was retrieved from the KEGG/GLYCAN database [20], and the annotations were retrieved from the CarbBank/CCSD database [21]. The tree structures obtained from the glycan data include 29 distinct node labels, while all the edge labels are omitted for simplicity. We evaluated the predictive performance of our tree kernels on the following two sets of glycan data, leukemia and cystic fibrosis. In the leukemia data set, the glycan structures annotated as *leukemic cells* were used as positive training examples, while those annotated as the other blood components *erythrocyte*, *serum*, and *plasma* without *leukemia* were used as negative training examples. The numbers of positive and negative data were respectively 191 and 289. In the cystic fibrosis data set, the glycan structures annotated as *cystic fibrosis* (a lethal genetic disorder affecting mainly the lungs and respiratory system) were used as positive training examples, while those annotated including the substring *bronch* and *respir* without *cystic fibrosis* were used as negative training examples. The numbers of positive and negative data were respectively 89 and 71.

For the cystic dataset, the overall trend in the results shown in Fig. 6 is similar to that for the XML dataset. Subpath kernel outperforms Vishwanathan kernel with any of the weighting schemes, which shows that a vertical path is effective as features. Especially, the exponential weighting performs the best. Vishwanathan kernel performs only slightly better than the Kailing kernel based on label nodes, although the number of node label is not so large and the size of trees is comparatively small in this dataset. This implies that subtrees are not effective features for this dataset. For the leukemia dataset, Vishwanathan kernel with the constant weight (CW) performs the best among all of the methods and the subtrees are effective as features, but Subpath kernel outperforms the linear-time kernel for the other two weighting schemes. From those results, we conclude Subpath kernel is competitive with Vishwanathan kernel in glycan classification tasks.

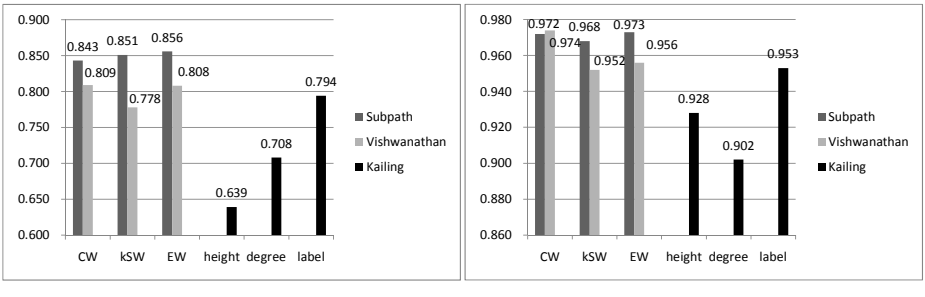


Fig. 6. Results for the cystic data (left) and the leukemia data (right) in the glycan classification task. The performances were measured in AUC with the 10-fold cross-validation.

4.3 Comparison of Execution Times of the Proposed Kernel and the Linear-Time Tree Kernel

In the previous subsections 4.1 and 4.2, we demonstrated that the proposed kernel (Subpath kernel) is competitive with the linear-time tree kernel (Vishwanathan kernel) in predictive performance by using real-world datasets. Here, we compare the execution time of Subpath kernel to that of Vishwanathan kernel on the three datasets. We calculated the gram matrices of the two kernels, and measured the average computation times needed for a single evaluation of a kernel function (by dividing the computation time of a whole gram matrix by the number of elements in the gram matrix). Figure 7 shows the average times of Subpath kernel and Vishwanathan kernel for the three datasets. The results show that Subpath kernel is consistently faster than Vishwanathan kernel, which means our kernel is quite efficient in practice despite of its worst case complexity.

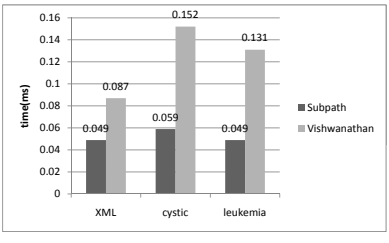


Fig. 7. Comparison of execution times of the proposed kernel and the linear-time kernel

5 Related Work

Since Haussler [5] introduced the framework of the convolution kernel, kernel functions for various kind of discrete structures, for example, strings [22,23,24], trees [6,7,1,8,10,9], and graphs [25,26] have been proposed.

The first tree kernel was proposed for parse trees by Collins and Duffy [6], and then it was generalized for labeled ordered trees [7,8], syntactic trees [10], and positional trees [9]. However, all of these kernels (explicitly or implicitly) exploit edge order information at each node in their definitions or algorithms, and therefore cannot be directly applied to unordered trees. For unordered trees, a hardness result for tree kernels using general tree-structured features was shown by Kashima [11]. Vishwanathan *et al.* [1] proposed an efficient linear-time kernel based on subtrees.

Another approach to complex-structured data is to use the pattern mining methods studied in the field of data mining [27,19]. They first enumerate substructure-patterns that frequently appear in datasets, and construct explicit feature vectors using the substructures. Since the number of substructures are enormous, it is not usually possible to construct the feature vectors within polynomial time, and various heuristic techniques for fast enumeration of substructures have been proposed.

6 Conclusion

In this paper, we proposed a new kernel for rooted labeled unordered trees based on tree subpath sets, and an efficient algorithm for computing the kernel function. The proposed algorithm is based on the multikey quicksort algorithm [13], and its computational complexity is $O((|T_1| + |T_2|)\log(|T_1| + |T_2|))$ on average, which is more efficient than the subgraph-based tree kernels using dynamic programming. We performed experiments on classification tasks, XML classification and glycan classification, and showed that the predictive accuracy of the proposed kernel was competitive with the existing unordered tree kernel [1], and is faster than the linear-time tree kernel in practice.

One of the possible future work is to further accelerate the proposed algorithm so that its worst case complexity is linear. Another direction is to allow mismatches of subpaths when computing the proposed kernels. To this goal, the techniques used in the mismatch string kernel [24] should be incorporated into the proposed algorithm.

References

1. Vishwanathan, S.V.N., Smola, A.: Fast kernels for string and tree matching. In: Advances in Neural Information Processing Systems, vol. 15, pp. 569–576 (2003)
2. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge (1999)
3. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
4. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
5. Haussler, D.: Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, UC Santa Cruz (1999)
6. Collins, M., Duffy, N.: Convolution kernels for natural language. In: Proceedings of the Fourteenth Annual Conference on Neural Information Processing Systems, pp. 625–632 (2001)
7. Kashima, H., Koyanagi, T.: Kernels for semi-structured data. In: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 291–298 (2002)
8. Kuboyama, T., Hirata, K., Aoki-Kinoshita, K.F., Kashima, H., Yasuda, H.: A gram distribution kernel applied to glycan classification and motif extraction. In: Proceedings of the Seventeenth International Conference on Genome Informatics, pp. 25–34 (2006)
9. Aiolli, F., Martino, G.D.S., Sperduti, A.: Route kernels for trees. In: Proceedings of the Twentie-sixth International Conference on Machine Learning, pp. 17–24 (2009)
10. Daumé III, H., Marcu, D.: A tree-position kernel for document compression. In: Proceedings of the Fourth Document Understanding Conference (2004)
11. Kashima, H.: Machine Learning Approaches for Structured-data. PhD thesis, Kyoto University (2007)
12. Ichikawa, H., Hakodaa, K., Hashimoto, T., Tokunaga, T.: Efficient sentence retrieval based on syntactic structure. In: Proceedings of the COLING/ACL, pp. 407–411 (2006)
13. Bentley, J.L., Sedgewick, R.: Fast algorithms for sorting and searching strings. In: Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 360–369 (1997)

14. Teo, C.H., Vishwanathan, S.V.N.: Fast and space efficient string kernels using suffix arrays. In: Proceedings of the Twentie-third International Conference on Machine Learning, pp. 929–936 (2006)
15. Shibuya, T.: Constructing the suffix tree of a tree with a large alphabet. IEICE Transactions on Fundamentals of Electronics 86(5), 1061–1066 (2003)
16. Kailing, K., Kriegel, H.P., Schönauer, S., Seidl, T.: Efficient similarity search for hierarchical data in large databases. In: Hwang, J., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 676–693. Springer, Heidelberg (2004)
17. Teo, C.H., Vishwanathan, S.V.N.: SASK: suffix arrays based string kernels (2006), http://users.cecs.anu.edu.au/~chteo/SASK.html
18. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), http://www.csie.ntu.edu.tw/~cjlin/libsvm
19. Zaki, M.J., Aggarwal, C.C.: Xrules: An effective structural classifier for xml data. Machine Learning Journal 62(1-2), 137–170 (2006)
20. Hashimoto, K., Hamajima, M., Goto, S., Masumoto, S., Kawashima, M., Kanehisa, M.: Glycan: The database of carbohydrate structures. Genome Informatics 14, 649–650 (2003)
21. Doubet, S., Albersheim, P.: Carbbank. Glycobiology 2(6), 505 (1992)
22. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. Journal of Machine Learning Research 2, 419–444 (2002)
23. Leslie, C., Eskin, E., Noble, W.: The spectrum kernel: A string kernel for SVM protein classification. In: Proceedings of the Pacific Symposium on Biocomputing, pp. 566–575 (2002)
24. Leslie, C., Eskin, E., Weston, J., Noble, W.S.: Mismatch string kernels for SVM protein classification. Neural Information Processing Systems 15, 1441–1448 (2003)
25. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 321–328 (2003)
26. Gärtner, T., Flach, P., Wrobel, S.: On graph kernels: Hardness results and efficient alternatives. In: Proceedings of the Sixteenth Annual Conference on Computational Learning Theory, pp. 129–143 (2003)
27. Washio, T., Motoda, H.: State of the art of graph-based data mining. SIGKDD Explorations 5(1), 59–68 (2003)

Classification Probabilistic PCA with Application in Domain Adaptation

Victor Cheng and Chun-Hung Li

Dept. of Computer Science
Hong Kong Baptist University
Kowloon Tong, Kowloon, HK
{victor,chli}@comp.hkbu.edu.hk
http://www.comp.hkbu.edu.hk/

Abstract. Conventional dimensionality reduction algorithms such as principle component analysis (PCA) and non-negative matrix factorization (NMF) are unsupervised. Supervised probabilistic PCA (SPPCA) can utilize label information. However, this information is usually treated as regression targets rather than discrete nominal labels. We propose a classification probabilistic PCA (CPPCA) which is an extension of probabilistic PCA. Unlike SPPCA, the label class information is turned into a class probabilistic function by using a sigmoidal function. As the posterior distribution of latent variables are non-Gaussian, we use Laplace approximation with Expectation Maximization (EM) to obtain the solution. The formulation is applied to a domain adaptation classification problem where the labeled training data and unlabeled test data come from different but related domains. Experimental results show that the proposed model has accuracy over conventional probabilistic PCA, SPPCA and its semi-supervised version. It has similar performance when compared with popular dedicated algorithms for domain adaptation, the structural correspondence learning (SCL) and its variants.

Keywords: dimensionality reduction, probabilistic PCA, domain adaptation, supervised projection.

1 Introduction

Many machine learning problems, such as image processing and text related mining, involve high dimensional data. Such data are often inefficiently processed in their original input spaces. The data can be projected to lower dimensional spaces so that they can be processed more efficiently in terms of computational time and space. Furthermore, hidden structures or features embedded in data can also be revealed. For example, latent semantic analysis (LSA)[5] in document analysis can capture synonymy information which is useful in document retrieving.

Conventionally, dimensionality reduction is done with LSA, PCA [9], probabilistic LSA [7], or probabilistic PCA [14]. These algorithms have two common features: they are unsupervised and the derived low dimensional space is usually

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 75–86, 2011.

corresponding to the directions of the original input space where the attributes of data have large variance. There are some other unsupervised approaches such as non-negative matrix factorization (NMF) [11]. These are classical unsupervised approaches in dimensionality reduction.

When label information of observations is available, dimensionality reduction should be guided by label information so that a better low dimensional space can be constructed to reflect this information. Linear discriminant analysis (LDA, Fisher's LDA) [13] can be used to produce a low dimensional space where the intra-class compactness and inter-class separation are accounted. However, it cannot incorporate unlabeled data in cases that labeled data are rare and expensive. Even semi-supervised LDA [3] is proposed to take the advantage of unlabeled data by incorporating the unlabeled data to regularization of LDA, this approach is still not suitable for applications such as domain adaptation [2], [1] where intra-covariance of data of different domains are highly concerned rather than inter-class separation or intra-class compactness considered in LDA. Some other related work including [10] also consider supervised dimensionality reduction from other perspectives.

Recently, supervised probabilistic PCA (SPPCA) [14] and its extension semi-supervised probabilistic PCA (S^2PPCA) [14] were proposed. In these approaches, the observed data, including its target values, and the latent variables are usually assumed to be linear dependant (with isotropic Gaussian noise) and hence the target values are treated as real values to be regressed. This assumption does not match the classification problems where target values are discrete and the loss should be the hinge loss. In this paper, we propose a classification probabilistic PCA (CPPCA) in which the class labels of training data are transformed to class conditional probabilities by using a sigmoidal function and thus hinge loss is implicitly employed. The introduction of this nonlinear function causes the model a little bit complex because the posterior probability distributions of latent variables are no longer Gaussian and their exact derivation and further processing are very difficult. We propose using Laplace approximation with expectation maximization (EM) to approximate these posterior distributions.

One of the applications of CPPCA is to tackle domain adaptation problems. Details will be given in Section 3. The rest of this paper is organized as following. In Section 2, the original probabilistic PCA is briefly reviewed and then followed by the formulation of CPPCA. Section 3 gives experimental results of CPPCA when applied to a domain adaptation problem. Finally, conclusions are presented in Section 4.

2 Classification Probabilistic PCA (CPPCA)

We follow the notation and style of [7] and [14], if possible, so that readers can follow all the work easily. Consider N_L labeled observations and N_U unlabeled observations denoted by M dimensional vectors $\{\mathbf{x_n}\}_{n=1}^{N_L+N_U}$, $\mathbf{x_n} \in \chi \subset \mathbb{R}^M$. Labels are binary values $\{0, 1\}$. Vectors of reduced dimensional space (i.e. vectors of latent variables) are denoted by $\mathbf{z} \in \mathcal{Z} \subset \mathbb{R}^K$, a K dimensional space with $K < M$.

2.1 Probabilistic PCA (PPCA) Revisited

In PPCA, dimensionality reduction is unsupervised and each observation $\mathbf{x}$ is generated as

$$\mathbf{x} = \mathbf{W_x z} + \mu_{\mathbf{x}} + \epsilon_{\mathbf{x}} \ , \tag{1}$$

where $\mathbf{W_x}$ is a $M \times K$ matrix and $\mu_x \in \mathcal{R}^M$. The vector $\mathbf{z}$ is Gaussian distributed with zero mean and unit variance, $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$. In (1), $\mathbf{z}$ plays as the vector of latent variables and noise $\epsilon_{\mathbf{x}}$ is assumed isotopic and Gaussian, i.e. $\epsilon_{\mathbf{x}} \sim \mathcal{N}(0, {\sigma_{\mathbf{x}}}^2\mathbf{I})$ in which the noise level depends on $\sigma_{\mathbf{x}}$. The parameters of the model thus are $\Omega = \{\mathbf{W_x}, \mu_{\mathbf{x}}, \sigma_{\mathbf{x}}^{\mathbf{2}}\}$. The values of Ω are optimized corresponding to the maximum log likelihood of all $\mathbf{x}$ which can be found by EM or computed explicitly. We quote the EM approach here because this approach will be used subsequently in later subsections. In the E-step, the posterior distribution of $\mathbf{z_n}$ is given by

$$\mathbf{z_n}|\mathbf{x_n} \sim \mathcal{N}(\mathbf{M^{-1}W_x^T}(\mathbf{x_n} - \mu_{\mathbf{x}}), \sigma_x^2\mathbf{M^{-1}}) \ , \tag{2}$$

where $\mathbf{M} = \mathbf{W_x^T W_x} + \sigma_x^2\mathbf{I}$. Hence, the expectation of posterior distribution of $\mathbf{z_n}$ given $\mathbf{x_n}$ is

$$< \mathbf{z_n}|\mathbf{x_n} >= \mathbf{M^{-1}W_x^T}(\mathbf{x_n} - \mu_{\mathbf{x}}) \ , \tag{3}$$

and the expectation $< \mathbf{z_n z_n^T}|\mathbf{x_n} >$ can be obtained by

$$< \mathbf{z_n z_n^T}|\mathbf{x_n} >= \sigma_{\mathbf{x}}^{\mathbf{2}}\mathbf{M^{-1}} + < \mathbf{z_n}|\mathbf{x_n} >< \mathbf{z_n}|\mathbf{x_n} >^{\mathbf{T}} \ , \tag{4}$$

where $< \cdot >$ denotes the expectation operator over the posterior distribution of $\mathbf{z_n}$.

For simplicity, $< \mathbf{z_n}|\mathbf{x_n} >$ will be denoted by $< \mathbf{z_n} >$ in the sequel, and it is assumed that the observations $\mathbf{x_n}$ have zero empirical mean, subtracting the empirical mean if not. In the M-step, parameters in Ω are updated to

$$\widetilde{\mu_x} = 0 \ , \tag{5}$$

$$\widetilde{\mathbf{W_x}} = \mathbf{X^T Z C^{-1}} \ , \tag{6}$$

and

$$\widetilde{\sigma_{\mathbf{x}}^{\mathbf{2}}} = \frac{1}{MN}\left[\sum_{n=1}^{N_u} \| \mathbf{x_n} \|^2 + \mathrm{tr}\left(\widetilde{\mathbf{W_x}}^{\mathbf{T}}\widetilde{\mathbf{W_x}}\mathbf{C}\right) - 2\mathrm{tr}\left(\widetilde{\mathbf{W_x}}\mathbf{Z^T X}\right)\right] \ , \tag{7}$$

where $\mathbf{X} = [\mathbf{x_1}, \ldots, \mathbf{x_{N_u}}]^T$, $\mathbf{Z} = [< \mathbf{z_1} >, \ldots, < \mathbf{z_{N_u}} >]^T$, $\mathbf{C} = \sum_{n=1}^{N_u} < \mathbf{z_n z_n^T} >$, $\| \cdot \|$ denotes the L^2 norm, and $\mathrm{tr}(\cdot)$ denotes the matrix trace. Iteration on EM will eventually gives converged parameters.

2.2 Classification Probabilistic PCA (CPPCA)

In this subsection, we consider the scenario that there are $N_L + N_U$ observations with the first N_L are labeled with $y_i \in \{0, 1\}$ where $i = 1, \ldots, N_L$. In order to incorporate the label information, the vector of latent variables $\mathbf{z_i}$ not only be responsible to the generation of the observations $\mathbf{x_i}$ but also effect the discrete values y_i. For unlabeled observations, $\mathbf{x_j}$, where $j = N_L + 1, \ldots, N_L + N_U$, is generated the same as that in the previously described PPCA. Since y_i is discrete while $\mathbf{z_i}$ is real valued, we propose using the sigmoidal function to transform the labels to class conditional probabilities $p(y_i = 1|\mathbf{z_i})$, and $p(y_i = 0|\mathbf{z_i}) = 1 - p(y_i = 1|\mathbf{z_i})$. As a result, the CPPCA generates $\mathbf{x}$ and y as follows.

$$\mathbf{x} = \mathbf{W_x z} + \mu_\mathbf{x} + \epsilon_\mathbf{x} \quad \text{, for all observations,} \tag{8}$$

$$p(y = 1|\mathbf{z}) = \phi(\mathbf{v^T z}) \quad \text{, for labeled observations,} \tag{9}$$

where $\epsilon_\mathbf{x} \sim \mathcal{N}(0, \sigma_\mathbf{x}^\mathbf{2}\mathbf{I})$, $\mathbf{v^T} \in \mathbb{R}^K$ is the transpose of the parameter vector $\mathbf{v}$ of the sigmoidal function

$$\phi(t) = \frac{1}{1 + e^{-t}} \ . \tag{10}$$

Thus the parameters of the model described by (8) and (9) are $\Omega = \{\mathbf{W_x}, \mu_\mathbf{x}, \mathbf{v^T}, \sigma_x^2\}$. The conditional distribution of $\mathbf{x}$ given $\mathbf{z}$ is the same as that in probabilistic PCA which is given by $\mathcal{N}(\mathbf{W_x z} + \mu_\mathbf{x}, \sigma_x^2\mathbf{I})$ and the prior of $\mathbf{z}$ is also $\mathcal{N}(0, \mathbf{I})$. The posterior of $\mathbf{z}$ depends on $\mathbf{x}$, and y if y is given. By applying the Bayes' theorem to the labeled observations,

$$p(\mathbf{z_n}|\mathbf{x_n}, y_n) \propto p(\mathbf{x_n}, y_n|\mathbf{z_n})p(\mathbf{z_n}), \quad \text{for } n = 1, \ldots, N_L \ . \tag{11}$$

From the model equations, $\mathbf{x}$ and y are independent given $\mathbf{z}$, hence,

$$p(\mathbf{z_n}|\mathbf{x_n}, y_n) \propto p(\mathbf{x_n}|\mathbf{z_n})p(y_n|\mathbf{z_n})p(\mathbf{z_n}) \quad \text{for } n = 1, \ldots, N_L \ . \tag{12}$$

For unlabeled observations,

$$p(\mathbf{z_n}|\mathbf{x_n}) \propto p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n}) \quad \text{for } n = N_L + 1, \ldots, N_L + N_U \ . \tag{13}$$

By referring to (2), $p(\mathbf{z_n}|\mathbf{x_n}) = \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$, where

$$\Sigma_{z|x} = \left[\frac{1}{\sigma_x^2}\mathbf{W_x^T W_x} + \mathbf{I}\right]^{-1}, \tag{14}$$

and

$$\mu_{z|x} = \Sigma_{z|x}\left[\frac{1}{\sigma_x^2}\mathbf{W_x^T}(\mathbf{x} - \mu_\mathbf{x})\right] \ . \tag{15}$$

On the other hand, finding $p(\mathbf{z_n}|\mathbf{x_n}, y_n)$ is not easy because $p(y_n|\mathbf{z_n})$ is sigmoidal and not Gaussian and hence $p(\mathbf{z_n}|\mathbf{x_n}, y_n)$ is not Gaussian. We propose using the

Laplace approximation to approximate $p(\mathbf{z_n}|\mathbf{x_n}, y_n)$ as Gaussian. With this approach, the approximated Gaussian mean is given by the maximum a posteriori (MAP) of the product of terms in the right hand side of (12) and the covariance matrix is given by the negative inverse of the Hessian matrix at MAP.

Let $\mathcal{L}$ denotes the logarithm of the right hand side terms of (12). After substituting (14) and (15) into $\mathcal{L}$ and dropping terms independent of $\mathbf{z_n}$, we get

$$\mathcal{L}(\mathbf{z_n}) = (\log\phi(\mathbf{v^T z_n}))^{y_n}(\log(1-\phi(\mathbf{v^T z_n})))^{1-y_n} - \frac{1}{2}(\mathbf{z_n} - \mu_{\mathbf{z}|\mathbf{x}})^{\mathbf{T}}\mathbf{\Sigma_{z|x}^{-1}}(\mathbf{z_n} - \mu_{\mathbf{z}|\mathbf{x}}). \tag{16}$$

Since the quadratic term (including the negative sign) in the above equation is concave and logarithm of a sigmoidal function is concave as well, $\mathcal{L}$ is concave and has a maximum. We use the Newton-Raphson method to find the maximum. Gradient of $\mathcal{L}$ and the Hessian matrix are given by

$$\nabla\mathcal{L} = (y_n - \phi(\mathbf{v^T z_n}))\mathbf{v^T} - \Sigma_{z_n|x}^{-1}(\mathbf{z} - \mu_{\mathbf{z}|\mathbf{x}}) \ \text{, and} \tag{17}$$

$$\nabla\nabla\mathcal{L} = \mathbf{H} - \Sigma_{z|x}^{-1} \ , \tag{18}$$

where

$$\mathbf{H} = -\phi(\mathbf{v^T z_n})(1 - \phi(\mathbf{v^T z_n}))\mathbf{v v^T} \ . \tag{19}$$

The Newton-Raphson iteration equation is then

$$\mathbf{z_n^{(t+1)}} = \mathbf{z_n^{(t)}} - (\mathbf{H} - \mathbf{\Sigma_{z|x}^{-1}})^{-1}\nabla\mathcal{L} \ . \tag{20}$$

Having found the maximum posterior $\mathbf{z_n^*}$, the Laplace approximation to $p(\mathbf{z_n}|\mathbf{x_n}, y_n)$ as Gaussian is given by

$$p(\mathbf{z_n}|\mathbf{x_n}, y_n) \approx \mathcal{N}(\mathbf{z_n}^*, (\mathbf{\Sigma}_{z|x}^{-1} - \mathbf{H})^{-1}) \ . \tag{21}$$

2.3 EM Learning for CPPCA

The parameters in Ω of CPPCA given by (8) and (9) can be learnt by EM. The E-step involves in finding the posterior distribution of $\mathbf{z_n}$ for each observation, observations with and without labels. Since all posterior distributions are now Gaussian, they can be identified by their own sufficient statistics, $< \mathbf{z_n} >$ and $< \mathbf{z_n z_n^T} >$. These sufficient statistics can be obtained from (14) and (15) for unlabeled observations, and (21) for labeled observations.

$$< \mathbf{z_n} >= \begin{cases} \mathbf{z_n}^* & \text{for } n = 1, \ldots, N_L \\ \mu_{\mathbf{z_n}|\mathbf{x_n}} & \text{for } n = N_L + 1, \ldots, N_L + N_U \end{cases} \tag{22}$$

$$< \mathbf{z_n z_n^T} >= \begin{cases} (\mathbf{\Sigma_{z|x}^{-1}} - \mathbf{H})^{-1} + < z_n >< z_n >^T & \text{for } n = 1, \ldots, N_L \\ \mathbf{\Sigma_{z_n|x_n}} + < z_n >< z_n >^T & \text{for } n = N_L + 1, \ldots, N_L + N_U \end{cases} \tag{23}$$

In the M-step, parameters in Ω can be updated by maximizing the expectation of the complete log likelihood $< \Psi >$ over the posterior distribution of $\mathbf{z_n}$, given by the following equation,

$$\begin{aligned} < \Psi >= \sum_{n=1}^{N_L} \int_{-\infty}^{\infty} p(\mathbf{z_n}|\mathbf{x_n}, y_n) \log(p(\mathbf{z_n})p(\mathbf{x_n}|\mathbf{z_n}) \cdot \\ p(y_n|\mathbf{z_n}))d\mathbf{z_n} + \sum_{n=N_L+1}^{N_L+N_U} \int_{-\infty}^{\infty} p(\mathbf{z_n}|\mathbf{x_n}) \cdot \\ \log(p(\mathbf{z_n})p(\mathbf{x_n}|\mathbf{z_n}))d\mathbf{z_n} \end{aligned} \tag{24}$$

By setting partial derivative with respect to parameters $\mathbf{W_x}, \mu_\mathbf{x}, \sigma_\mathbf{x}^\mathbf{2}$ in Ω to zero, the following update equations are obtained.

$$\widetilde{\mu_x} = 0 \ , \tag{25}$$

$$\widetilde{\mathbf{W_x}} = (\mathbf{X_L^T Z_L} + \mathbf{X_U^T Z_U})(\mathbf{C_L} + \mathbf{C_U})^{-1} \ , \tag{26}$$

$$\widetilde{\sigma_\mathbf{x}^\mathbf{2}} = \frac{1}{MN} \left[\sum_{n=1}^{N_u} \| \mathbf{x_n} \|^2 + \text{tr}\left(\widetilde{\mathbf{W_x}}^\mathbf{T} \widetilde{\mathbf{W_x}} (\mathbf{C_L} + \mathbf{C_U}) \right) - 2\text{tr}\left(\widetilde{\mathbf{W_x}} (\mathbf{Z_L^T X_L} + \mathbf{Z_U^T X_U}) \right) \right], \tag{27}$$

where $\mathbf{C_L}, \mathbf{C_U}, \mathbf{Z_L}, \mathbf{Z_U}, \mathbf{X_L}, \mathbf{X_U}$ are defined as that in (7), noting for labeled and unlabeled observations as indicated in the subscript. For updating $\mathbf{v^T}$ in the M-step, setting the derivative of $< \Psi >$ with respect to $\mathbf{v^T}$ to zero results in a nonlinear equation in $\mathbf{v^T}$.

$$\sum_{n=1}^{N_L} \left(y_n - \phi(\mathbf{v^T z_n}) \right) \mathbf{z_n} = 0 \ . \tag{28}$$

Again, this equation can be solved easily with Newton-Raphson method. The Jacobian matrix $\mathbf{J}$ of left hand side of (28) is

$$\mathbf{J} = \sum_{n=1}^{N_L} -\phi(\mathbf{v^T z_n})(1 - \phi(\mathbf{v^T z_n}))\mathbf{z_n z_n^T} \ , \tag{29}$$

and hence the iteration equation for $\mathbf{v^T}$ is given by

$$\mathbf{v^{T(t+1)}} = \mathbf{v^{T(t)}} - \mathbf{J}^{-1} \left[\sum_{n=1}^{N_L} \left(y_n - \phi(\mathbf{v^T z_n}) \right) \mathbf{z_n} \right] \tag{30}$$

This EM iteration continues until the change of Ω between two consecutive EM iterations below a user specified threshold. After the EM learning processes, arrival of unseen unlabeled observation $\hat{\mathbf{x}}$ will be mapped to latent variable $\hat{\mathbf{z}}$ having Gaussian distribution identified by (14) and (15). For an unseen labeled observation $(\hat{\mathbf{x}}, \hat{y})$, $\hat{\mathbf{z}}$ is given by (21).

3 Experimental Results

In this section, we describe the application of the CPPCA to domain adaptation of classification problems. Traditionally, many statistical learning algorithms assume that training data and test data come from an identical and independent distribution (iid). Unfortunately, this assumption is often inappropriate. Owing to many factors, such as sampling techniques, time changes, etc., training data and test data may have different statistical properties. In domain adaptation problems, the difference is due to training data and test data coming from different but related domains. This problem is very common in natural language processing. For example, we may have a large corpus from one domain and large effort and resources has been spent on annotating it. We would like to use the results to analyze corpora coming from other domains.

With training data and test data coming from different domains, learning a classifier such as support vector machine (SVM) [4] with the training data and classifying the test data may have degarded results. Sample re-weighting [15] is one of the popular approaches for domain adaptation. For example, Huang [8] found the weights for instances of samples by minimizing the maximum mean discrepancy (MMD) [6]. This approach, however, cannot be directly applied to high dimensional data with sparse attributes. Some attributes may appear only in training data while some others appear only in test data and hence minimizing MMD is not effective. Another approach for this problem is to project both training data and test data to a common feature structure [1] or a low dimensional latent space followed by minimizing the MMD or embed this procedure in finding the low dimensional space as proposed by Pan [12]. These two approaches actually assume that there is a common space where the discrepancy of distributions of training and test data is minimized.

An intuitive approach for domain adaptation without minimizing MMD explicitly is to stack the training and test data together and then perform a probabilistic PCA to find the common latent random variables that generate both training and test data. Since now all the data are assumed to be generated from one set of latent random variables, the MMD is automatically minimized in the latent space formed by these variables. The success of this approach depends on whether the stacked data can be represented by these random variables with low error. For domain adaptation of classification task, class labels are usually available. This information can be incorporated by the proposed CPPCA rather than probabilistic PCA so that the low dimensional space found is prevalent to the task. We tested this approach with some experiments. Owing to the limited space, we report the one having larger scale.

3.1 Product Review Adaptation Experiment

This experiment deals with an Amazon product review dataset[1] [1] of four different types of products: books, DVDs, electronic appliances, and kitchen appliances. Each review has a rating, 0 - 5 stars, given by a reviewer. In the experiment, reviews with more than three stars were labeled as positive reviews and those with less than three stars were regarded as negative reviews. Three stars reviews were discarded because their sentiments were ambiguous. Unigram and bigram features were used to form the bag-of-words representation. Since, there were over a million of features, we simply removed less frequent words and then removed reviews with zero words so that 5834 features were left. Table 1 gives a brief summary for this dataset.

Table 1. Summary of the Amazon product review dataset

	Books	DVDs	Electronic appliances	Kitchen appliances
Number of unlabeled patterns	4445	3555	5677	5936
Number of labeled patterns	1992	1992	1998	1992

In order to compare our results with that of Blitzer et al. [1], we followed their experimental setup. For each domain, 1600 labeled patterns were selected as the training set, and the rest, about 400 labeled patterns, formed the test set. The label information of the test set was solely used for accuracy evaluation and would not be used by any algorithms. The test set and the unlabeled patterns of the domain would be mixed to form the unlabeled set. The goal of the experiment was to try to use the training set of one domain to predict the labels of the test sets of other domains as accurately as possible. For example, in a domain adaptation of Books domain to DVDs domain, 1600 labeled patterns of book reviews formed the labeled training set. All the unlabeled patterns of book reviews (i.e. 4445 book patterns) were mixed with 392 labeled patterns of DVDs to form the unlabeled set and the task was try to predict the labels of these 392 DVDs patterns accurately. The first part of this experiment is to study the performance of in-domain classification, both the training set and test set coming from the same domain. The data was firstly projected to a 100 dimensional space with different projection algorithms and followed by classification done with a logistic classifier. The baseline was the accuracy obtained with a logistic classifier working on the original input space. Results are illustrated in Table 2. Classification accuracies shown in the table are the means obtained with five-fold cross-validation.

The results in Table 2 show that classification of the dataset in a projected lower dimensional space has some accuracy loss, compared to the baseline logistic classifier. PPCA and S^2PPCA have nearly the same results indicating that the label information of the training set is not properly exploited by S^2PPCA.

[1] The dataset can be found at http://www.cs.jhu.edu/ mdredze/datasets/sentiment/

Table 2. In-domain classification accuracy of the Amazon product review dataset

Domains	Logistic classifier (baseline)	PPCA + Logistic classifier	S^2PPCA + Logistic classifier	CPPCA + Logistic classifier
Books	0.80	0.80	0.79	0.82
DVDs	0.82	0.79	0.80	0.83
Electronic appliances	0.84	0.82	0.82	0.83
Kitchen appliances	0.88	0.85	0.84	0.88

On the other hand, CPPCA has better performance because the hinge loss is used. CPPCA has accuracy similar to the baseline logistic classifier. This implies almost all the required classification information is preserved during projections. For the Books domain and DVDs domain, CPPCA even has better accuracy. This is possible because there are additional unlabeled patterns involved in the projections with CPPCA.

The second part of the experiment is concerned with domain adaptation, training set and test set coming from different domains. The results are shown in Figure 1 and 2. In these figures, the thick horizonal line indicates the accuracy of in-domain classification for reference. The name of domains are abbreviated as "B" for Books, "D" for DVDs, "E" for Electronic appliances, and "K" for Kitchen appliances. Moreover, "B → D" means domain adaptation from the Books domain to the DVDs domain, and so on.

From the figures, projection with PPCA and S^2PPCA followed by logistic classification has only little advantage over classification in the original input space. CPPCA works better. Comparing with the baseline classification, domain adaptation with CPPCA has significant improvements, about 0.03-0.06 increase in accuracy, for D→B, E→B, E→D and B→E. Overall average increase in accuracy is 0.022. Although the average improvement seems small, the performance of CPPCA is not bad. Firstly, the Kitchen appliances domain and the Electronic

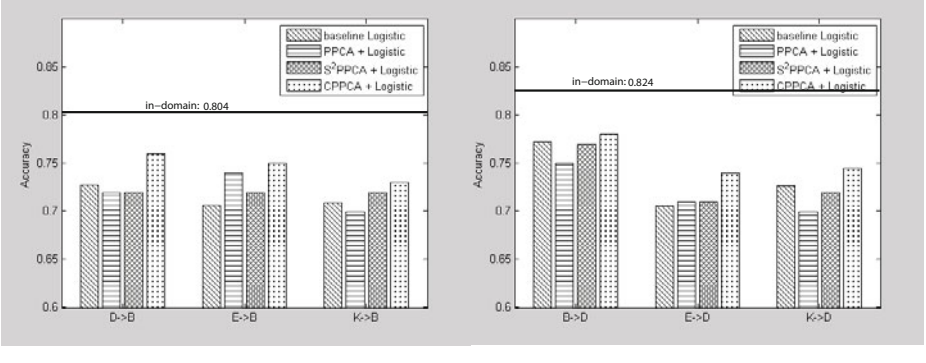


Fig. 1. Classification accuracy of domain adaptation. Left: from other domains to Books (B), right: from other domains to DVDs (D).

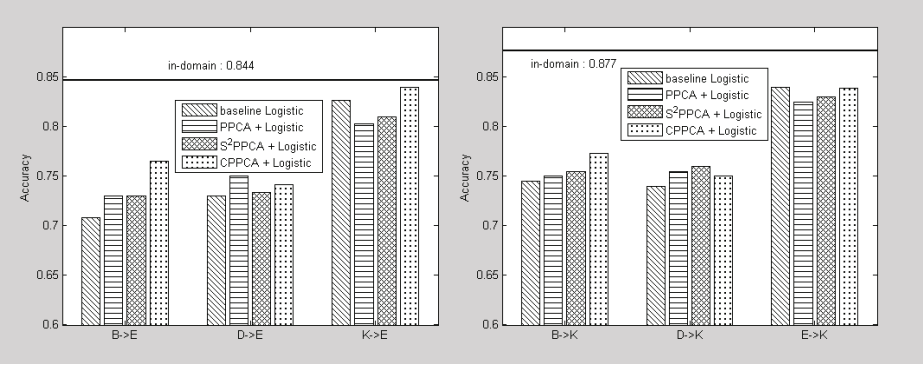


Fig. 2. Classification accuracy of domain adaptation. Left: from other domains to Electronic appliances (E), right: from other domains to Kitchen appliances (K).

appliances domain are very close to each other. It is because many appliances in a kitchen are electronic appliances. As a result, classification accuracy of K→E and E→K are quite close to the in-domain accuracy even no domain adaptation. The Books domain and the DVDs domain are also believed to be close to each other.

Finally, we compare our CPPCA results with the quoted results of Blitzer's work [1], which employed the structural correspondence learning (SCL) [2] domain adaptation algorithm and its variant SCL-MI. Since SCL and SCL-MI used both the attributes of projected spaces and attributes of the original space, we augmented our CPPCA projected patterns with the original input attributes for fairness. The results are illustrated in Figure 3 and 4.

Comparing to baseline accuracy, the average improvements are 0.023, 0.035, and 0.029 for SCL, SCL-MI, and augmented CPPCA, respectively. The accuracy of SCL-MI for K→E exceeds in-domain accuracy because unlabeled patterns

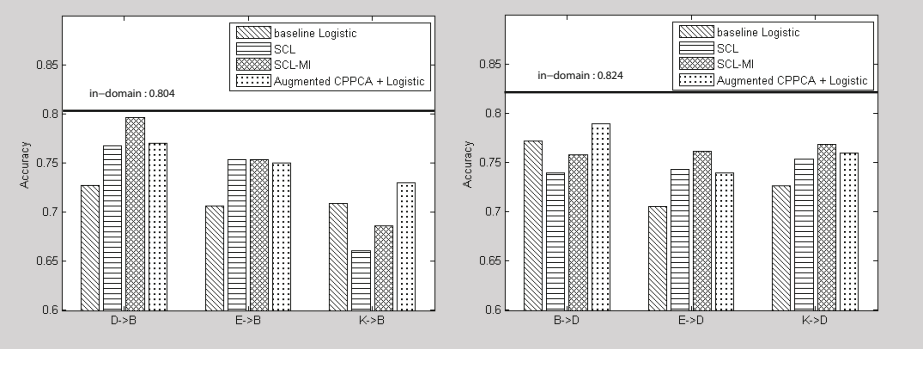


Fig. 3. Classification accuracy of domain adaptation. Left: from other domains to Books (B), right: from other domains to DVDs (D).

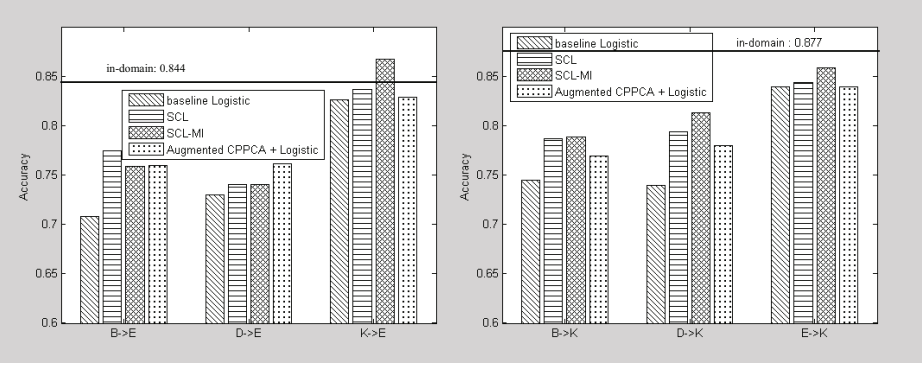


Fig. 4. Classification accuracy of domain adaptation. Left: from other domains to Electronic appliances (E), right: from other domains to Kitchen appliances (K).

were used in SCL-MI while they were ignored in in-domain classification. By comparing the results of CPPCA in Figure 1 and 2 and augmented CPPCA in Figure 3 and 4, it can be seen that augmented CPPCA has average accuracy gain 0.007 only. This shows that CPPCA preserves almost all useful information in projection, and augmenting the original input attributes to it does not benefit much. As a result, CPPCA is very suitable for dimensionality reduction in cases that maintaining classification accuracy is important. In above experiments, we only projected the patterns to 100 dimensional spaces. It is because the performance of CPPCA for this dataset is quite stable when the projected dimension is above 100.

4 Conclusions

Many data mining problems are high dimensional. Projecting the data to a much lower dimensional space enables the saving of temporal and spatial cost. Classical unsupervised dimensionality reduction algorithms such as PPCA and NMF cannot utilize label information dimensionality reduction. In this paper, we propose CPPCA to incorporate discrete label information of a classification task in deriving a low dimensional projected space. By stacking the training data and test data followed by using CPPCA, domain adaptation can be achieved at low dimensional latent spaces. Experimental results show that this approach has performance advantage over PPCA and S^2PPCA in terms of accuracy. In-domain experiments also show that nearly all useful classification information can be preserved during projection with CPPCA. Finally, we would like to mention that domain adaptation is not the only application for CPPCA. It can be applied to other problems where PPCA can be applied.

Acknowledgement

This work is partially supported by HKBU research grant FRG2/08-09/103.

References

1. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: Annual Meeting-Association for Computational Linguistics, pp. 440–447 (2007)
2. Blitzer, J., Mcdonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia, pp. 120–128. Association for Computational Linguistics (July 2006)
3. Cai, D., He, X., Han, J.: Semi-supervised discriminant analysis. In: IEEE 11th International Conference on Computer Vision, ICCV 2007, pp. 1–7 (2007)
4. Cristianini, N., Shawe-Taylor, J.: An introduction to support vector machines: and other kernel-based learning methods, 1st edn. Cambridge University Press, Cambridge (March 2000)
5. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41, 391–407 (1990)
6. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.J.: A kernel method for the two-sample problem. CoRR, abs/0805.2368 (2008)
7. Hofmann, T.: Probabilistic latent semantic analysis. In: Proc. of Uncertainty in Artificial Intelligence, UAI 1999, Stockholm (1999)
8. Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In: Advances in Neural Information Processing Systems, vol. 19, pp. 601–608 (2007)
9. Jolliffe, I.T. (ed.): Principal Component Analysis. Springer, Heidelberg (2002)
10. Kim, H., Howland, P., Park, H.: Dimension reduction in text classification with support vector machines. J. Mach. Learn. Res. 6, 37–53 (2005)
11. Lee, D.D., Sebastian Seung, H.: Algorithms for non-negative matrix factorization. In: NIPS, pp. 556–562 (2000)
12. Pan, S.J., Kwok, J.T., Yang, Q.: Transfer learning via dimensionality reduction. In: Proc. of the Twenty-Third AAAI Conference on Artificial Intelligence (2008)
13. Welling, M.: Fisher linear discriminant analysis, http://www.ics.uci.edu/~welling/classnotes/papers_class/Fisher-LDA.pdf [Online; accessed August 15, 2010]
14. Yu, S., Yu, K., Tresp, V., Kriegel, H.-P., Wu, M.: Supervised probabilistic principal component analysis. In: Ungar, L. (ed.) 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 464–473. ACM Press, New York (August 2006)
15. Zadrozny, B.Z.: Learning and evaluating classifiers under sample selection bias. In: International Conference on Machine Learning ICML 2004, pp. 903–910 (2004)

Probabilistic Matrix Factorization Leveraging Contexts for Unsupervised Relation Extraction

Shingo Takamatsu[1], Issei Sato[2], and Hiroshi Nakagawa[2]

[1] Sony Corporation, 5-1-12 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
Shingo.Takamatsu@jp.sony.com
[2] University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033 Japan
sato@r.dl.itc.u-tokyo.ac.jp, n3@dl.itc.u-tokyo.ac.jp

Abstract. The clustering of the semantic relations between entities extracted from a corpus is one of the main issues in unsupervised relation extraction (URE). Previous methods assume a huge corpus because they have utilized frequently appearing entity pairs in the corpus. In this paper, we present a URE that works well for a small corpus by using word sequences extracted as relations. The feature vectors of the word sequences are extremely sparse. To deal with the sparseness problem, we take the two approaches: dimension reduction and leveraging context in the whole corpus including sentences from which no relations are extracted. The context in this case is captured with feature co-occurrences, which indicate appearances of two features in a single sentence. The approaches are implemented by a probabilistic matrix factorization that jointly factorizes the matrix of the feature vectors and the matrix of the feature co-occurrences. Experimental results show that our method outperforms previously proposed methods.

Keywords: Unsupervised Relation Extraction, Probabilistic Matrix Factorization, Dimension Reduction.

1 Introduction

Extracting semantic relations between entities from texts is one of the main tasks in text mining, and the unsupervised approach can find unknown relations without human input and is widely applicable to new domains. Unsupervised relation extraction (URE) can be viewed as a combination of two steps: extracting relational tuples from texts, and clustering those that have synonymous relations. (The tuples consist of two entities and a word sequence that indicate the relation between them.) This clustering step involves the main issue of URE. The tuples that mean "collaboration between two musicians M1 and M2", for example, are described with different word sequences, such as (M1, *collaborated with*, M2), (M1, *worked alongside*, M2) or (M1, *partnered with*, M2). We want to cluster such tuples into a single cluster which indicates the "collaboration" relation.

In existing URE methods [5,8,14] synonymous tuples are found by looking for tuples having the same pair of entities. Tuples with the same entity pair, however,

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 87–99, 2011.

do not necessarily have the same meaning. Moreover, multiple appearances of the same entity pair are so rare in a target corpus that the exiting methods need a huge amount of corpus data (in [14], e.g., 2.1 million tuples) and the extracted relations tend to be general. These methods are thus not suitable when we are interested in a specific domain, such as a long tail one or a technical one in which a relatively small amount of text is available, and want to extract domain-specific relations.

We have therefore developed a URE that works well for a small specific corpus in which almost all the entity pairs of tuples appear only once. Our URE doesn't use the frequently appearing entity pairs used by the existing URE methods. We extract features of the tuples mainly from the word sequences. Because only a few features are extracted from the word sequence, however, we have sparse feature vectors with elements that are mostly zeros. It is widely known that clustering sparse vectors does not work well.

Our approach to the sparseness is dimension reduction, which projects the original feature vectors in a high-dimensional space to compressed feature vectors in a lower-dimensional space and consequently mitigates the sparseness. Directly applying a standard dimension reduction such as Latent Semantic Indexing (LSI) [3] to feature vectors of tuples does not work well, however, because the feature vectors of tuples are sparser than those of targets that often used in natural language processing (e.g., documents).

Our solution is to leverage contexts in the whole corpus together with dimension reduction. Unlike existing URE methods, our URE utilizes the sentences from which a tuple is not extracted. A part of features are extracted from words, and the features appear not only in tuples but in contexts of such sentences. The contexts have information on meanings of such features [4] and thus help to estimate the compressed feature vectors that represent meanings of the tuples. The contexts in this case are captured with feature co-occurrences, which indicate that two features appear in a single sentence, and are leveraged by a probabilistic matrix factorization (PMF) that jointly factorizes the feature vector matrix (each row of which is the feature vector) and the feature co-occurrence matrix (each element of which is the frequency of feature co-occurrences). A part of estimated parameters of the PMF are used as the compressed feature vectors. We call this PMF *context-leveraged* PMF (CL-PMF) and experimentally show that the feature vectors compressed by CL-PMF are clustered with higher purity than those compressed by existing dimension-reduction methods.

2 Related Work

Our dimension reduction with matrix factorization is related to LSI [3], which uses singular value decomposition (SVD) to factorize a document-term matrix into low rank matrices and obtains a low-rank approximation. The row vector of the left/right singular matrix represents a latent concept (a large portion of information) of the corresponding document/term and is therefore used as its compressed vector in the low-dimensional space. Analysis tasks such as document

clustering and word sense disambiguation can be performed with the compressed vectors.

Dimension-reduction methods for semantic relations between entities were proposed in [13]. In that work, entities (assumed to be general nouns) were extended with a thesaurus and queried to a search engine to gather features. The resultant feature vectors are compressed with LSI. This work shows the effectiveness of a dimension reduction. The method uses rich linguistic resources not expected in a specific domain, whereas our method uses the feature co-occurrences in the target corpus, which are naturally obtained.

Word sense disambiguation using word co-occurrences was proposed in [11]. Each element of a word co-occurrence matrix is the frequency with which two words appear in a single sentence in a corpus. The similarities between meanings of words can be calculated with the corresponding row vectors of the matrix. The vectors are compressed with LSI, and the compressed vector, which represents the latent concept of the word, is used for the word sense disambiguation. In a similar way, CL-PMF uses a feature co-occurrence matrix to estimate the latent concepts of features (this is explained in Section 3.3.1).

Joint matrix factorizations related to CL-PMF are the link-content matrix factorization using link information for Web page classification [15] and the joint matrix factorization used with collaborative filtering using social connections to make recommendations [6]. While these methods model two matrices of the same size, in our model one matrix is bigger than the other. As explained in Section 3.3.2, this size difference plays an important role to exploit kinds of features that appear not in extracted tuples but in the target corpus. In addition, we introduce a full Bayesian approach, which are reported to boost the performance of matrix factorization [10].

3 Unsupervised Relation Extraction

The outline of our URE is as follows: First we discover relations from texts and extract relational tuples (Section 3.1). Then we extract features from the tuples and construct the feature vectors (Section 3.2). We then use CL-PMF to compress the feature vectors into a low-dimensional space (Section 3.3). Finally, we use a conventional clustering method to cluster the compressed feature vectors.

3.1 Relation Discovery

The relations that we want to extract take the form of a tuple $t = (e_1, s, e_2)$, where e_1 and e_2 indicate entities, and s is a word sequence that indicates the relation between them. The following process for extracting tuples was inspired by the *full-scale* relation extraction in [1,2].

The sentences in the target corpus are first parsed with a dependency parser to obtain their dependency graph representations. For the parsed sentences, the system finds all the named entities, which are used as entities, e_1 or e_2. When a sentence contains more than two entities, the system detects the *dependency*

path (path of the dependency graph) connecting two entities. The system then extracts tuples when the dependency path from e_1 to e_2 does not contain a sentence-like boundary, such as a relative clause, and the number of words between e_1 and e_2 is less than 10. The word sequence s is extracted as the part of the sentence based on the parse structure between e_1 and e_2. Two examples of extracted tuples are (*Sasha Allen, has also toured with, Alicia Keys*) and (*Kings Cross, shared the stage with, Kasabian*).

3.2 Feature Extraction

The following three kinds of features are extracted from a tuple.

Unigram Feature: A unigram feature, which is extracted from a word, is a pair consisting of the word's stemmed form and part of speech. We extract this feature from each word in the word sequence s. The unigram features of stop words (common words that are filtered out prior to a processing) on the dependency path from e_1 to e_2 are used to capture the word's impact (e.g., "*studied*" and "*studied at*"), and stop words not on the path are filtered out.

Bigram Feature: A bigram feature is a conjunction of the two unigram features alongside the dependency path from e_1 to e_2.

Entity Tag Feature: An entity tag feature consists of named entity tags for the two entities. The tags are tagged by a named entity tagger. The used tags are PERSON, LOCATION, and ORGANIZATION.

We use a vector space model to represent a tuple with these features. An element of a feature vector is a frequency of the corresponding feature, and a large number of kinds of features are extracted from tuples in a target corpus. The feature vector of a tuple is therefore high-dimensional and sparse.

3.3 CL-PMF for Dimension Reduction

Here we first present an intuitive description of CL-PMF. Then we define a feature co-occurrence matrix, describe CL-PMF, and describe its inference.

3.3.1 Intuitive Description of CL-PMF

Let $\mathbf{X} = \{x_{ij}; i = 1, \ldots, N, j = 1, \ldots, M\}$ be a feature vector matrix, where the row vector $\mathbf{x}_i \in \mathcal{R}^M$ corresponds to the feature vector of the i-th tuple, N is the number of tuples, and M is the number of kinds of features for $\mathbf{X}$. Under the assumptions of linear transform, dimension reduction can be expressed as the matrix factorization,

$$\mathbf{X} \approx \mathbf{U}^T\mathbf{V}, \tag{1}$$

where we define $D(D < M)$ as the number of dimensions of a low dimensional space, $\mathbf{U}$ is the $D \times N$ matrix in which the column $\mathbf{U}_i \in \mathcal{R}^D$ corresponds to the compressed feature vector of the i-th tuple, and $\mathbf{V}$ is the $D \times N$ matrix in which the column $\mathbf{V}_j \in \mathcal{R}^D$ corresponds to the compressed vector representing the

latent concept of the j-th kind of feature. This matrix factorization represents a low rank approximation like LSI. In our task, matrix $\mathbf{U}$ and matrix $\mathbf{V}$ are not well estimated because the feature vector matrix $\mathbf{X}$ is sparse in that each row $\mathbf{x}_i$ is sparse as explained in Section 3.2 and the number of the rows of $\mathbf{X}$ (i.e., the number of feature vectors) is small because the appearances of tuples are not frequent in a small corpus.

To resolve the shortage of the feature vectors we utilize the contexts of all sentences in the corpus. We use a feature vector matrix of sentences $\mathbf{X}_s$, where each row is the feature vector of each sentence in the corpus. The features of a sentence are unigram features extracted from words in the sentence. Note that the unigram features used in $\mathbf{X}$ is also used in $\mathbf{X}_s$. Consider the following matrix factorization:

$$\mathbf{X}_s \approx \mathbf{U}_s^T \mathbf{V}_s, \tag{2}$$

where $\mathbf{U}_s$ and $\mathbf{V}_s$ are defined in the same way as they are in Eq. (1). In this case, $\mathbf{V}_s$ is well estimated because we have an enough number of the sentences. Now, we regard a tuple as a short sentence and use $\mathbf{V}_s$, which represents the latent concepts of features estimated by the sentences, for the tuples. To utilize $\mathbf{V}_s$ for $\mathbf{V}$, we combine the two matrix factorizations by sharing the columns corresponding to the same unigram features in $\mathbf{V}$ and $\mathbf{V}_s$. However, we don't need $\mathbf{U}_s$ in Eq. (2). For a compact parameter representation, we calculate as follows:

$$\mathbf{X}_s^T \mathbf{X}_s \approx \left(\mathbf{U}_s^T \mathbf{V}_s\right)^T \mathbf{U}_s^T \mathbf{V}_s = \mathbf{V}_s^T \left(\mathbf{U}_s^T \mathbf{U}_s\right) \mathbf{V}_s = \mathbf{V}_s^T \mathbf{V}_s, \tag{3}$$

where we assume $\mathbf{U}_s^T \mathbf{U}_s = \mathbf{I}$ ($\mathbf{I}$ is the identity matrix), using a freedom of the matrix factorization in Eq. (2) and for example, SVD meets the assumption. We use $\mathbf{X}_s^T \mathbf{X}_s$ instead of $\mathbf{X}_s$ and consequently we do not have to consider $\mathbf{U}_s$. $\mathbf{X}_s^T \mathbf{X}_s$ is a feature co-occurrence matrix, where each element is a frequency of corresponding feature co-occurrences. We combine the two matrix factorizations in Eq. (1) and Eq. (3). Since the shared columns of $\mathbf{V}$ are well estimated, we expect $\mathbf{U}$ and the rest of $\mathbf{V}$ (i.e., the columns of bigram features and entity tag features) to be well estimated, too.

3.3.2 Feature Co-occurrence Matrix

We incorporate bigram features and entity tag features in $\mathbf{X}_s^T \mathbf{X}_s$ and define the feature co-occurrence matrix $\mathbf{F} = \{f_{jk}; j = 1, \ldots, L, k = 1, \ldots, L\}$, where $L (L \geq M)$ is the number of kinds of features for $\mathbf{F}$. $\mathbf{F}$ is a symmetric matrix and both rows and columns of $\mathbf{F}$ correspond to kinds of features. The first M kinds of features of $\mathbf{F}$ are same as those of $\mathbf{X}$ and the rest kinds of features correspond to kinds of unigram features not in any of tuples (i.e., kinds of unigram features not used in $\mathbf{X}$, but used in $\mathbf{X}_s^T \mathbf{X}_s$). Note that L is greater than M because to capture the contexts in the whole corpus, we additionally use new kinds of unigram features not in any of the tuples. An element corresponding to two unigram features describes a frequency with which the two unigram features

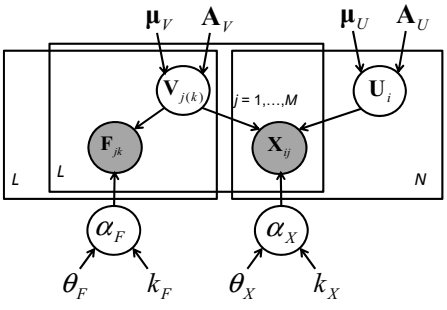


Fig. 1. Graphical model of CL-PMF

appear in a single sentence. For the count, all the sentences in the target corpus are used. Elements not corresponding to two unigram features (i.e., elements related to bigram features and entity tag features) are set at zero. The size of $\mathbf{V}$ is increased to L with the increase of kinds of features. Here are some samples of the feature co-occurrences that extracted from the sentence "*Spector had recently produced his solo single*": (had/VB, recent/NN), (had/VB, produc /VB), (solo/JJ, singl/NN).

We apply the weighting function $W(x) = \log(1 + x)$ [7] to the values of the elements of the data matrices, $\mathbf{X}$ and $\mathbf{F}$.

3.3.3 Context-Leveraged PMF

CL-PMF models the combination of the matrix factorizations in Eq. (1) and Eq.(3). CL-PMF is a probabilistic model over observed elements in $\mathbf{X}$ and $\mathbf{F}$ with Gaussian observation noise. A zero value in the matrices is interpreted to be un-observed. We use a full Bayesian approach [10].

CL-PMF is given by:

$$p(\mathbf{X},\mathbf{F}|\mathbf{U},\mathbf{V},\alpha_X,\alpha_F) = \prod_{i=1}^{N}\prod_{j=1}^{M}[p(x_{ij}|\mathbf{U}_i,\mathbf{V}_j,\alpha_X)]^{I_{ij}^X} \prod_{j=1}^{L}\prod_{k=1}^{L}[p(f_{jk}|\mathbf{V}_j,\mathbf{V}_k,\alpha_F)]^{I_{jk}^F},$$

$$p\left(x_{ij}|\mathbf{U}_i,\mathbf{V}_j,\alpha_X\right) = N\left(x_{ij}|\mathbf{U}_i^T\mathbf{V}_j,\alpha_X\right),$$

$$p\left(f_{jk}|\mathbf{V}_j,\mathbf{V}_k,\alpha_F\right) = N\left(f_{jk}|\mathbf{V}_j^T\mathbf{V}_k,\alpha_F\right),$$

where I_{ij}^X is the indicator variable of $\mathbf{X}$ which is equal to 1 if an element x_{ij} is observed and is equal to 0 otherwise, I_{jk}^F is the indicator valuable of $\mathbf{F}$, $N(x|\mu,\alpha)$ denotes the Gaussian distribution with mean μ and precision (inverse of variance) α, α_X is the precision of all the elements of $\mathbf{X}$, and α_F is the precision of all the elements of $\mathbf{F}$. The graphical model for CL-PMF is shown in Fig. 1. Both $\mathbf{X}$ and $\mathbf{F}$ are conditioned on $\mathbf{V}$. This corresponds to the share of $\mathbf{V}$ between the two matrix factorizations explained in Section 3.3.1.

The prior distributions are as follows:

$$p\left(\mathbf{U}|\boldsymbol{\mu}_U, \mathbf{A}_U\right) = \prod_{i=1}^{N} p\left(\mathbf{U}_i|\boldsymbol{\mu}_U, \mathbf{A}_U\right) \quad \text{and} \quad p\left(\mathbf{U}_i|\boldsymbol{\mu}_U, \mathbf{A}_U\right) = N\left(\mathbf{U}_i|\boldsymbol{\mu}_U, \mathbf{A}_U\right),$$
$$p\left(\alpha_X|k_X, \theta_X\right) = Gam\left(\alpha_X|k_X, \theta_X\right),$$

where $\boldsymbol{\mu}_U$ and $\mathbf{A}_U$ are the mean and the precision of the Gaussian distribution and $Gam(x|k, \theta)$denotes the Gamma distribution with shape parameter k and scale parameter θ. Because of the symmetry of the model, the prior distribution of $\mathbf{V}$ is identical to that of $\mathbf{U}$. The prior of α_F is same as the prior of α_X. These priors are conjugate priors for the model described above.

Given the data and the prior distributions, we can now write the posterior distribution, which is the distribution of parameters conditioned on observed data and hyper parameters. The posterior distribution is given by

$$p\left(\mathbf{U}, \mathbf{V}, \alpha_X, \alpha_F, |\mathbf{X}, \mathbf{F}, \boldsymbol{\Psi}\right) \propto p\left(\mathbf{U}, \mathbf{V}, \alpha_X, \alpha_F|\boldsymbol{\Psi}\right) p\left(\mathbf{X}, \mathbf{F}|\mathbf{U}, \mathbf{V}, \alpha_X, \alpha_F\right),$$

where $\boldsymbol{\Psi} = \{\boldsymbol{\mu}_U, \boldsymbol{\mu}_V, \mathbf{A}_U, \mathbf{A}_V, k_X, k_F, \theta_X, \theta_F\}$ is the set of hyper parameters of the prior distributions. We use the expectation of $\mathbf{U}_i$ over the posterior distribution as the compressed feature vector of the i-th tuple.

3.3.4 Inference

We use Gibbs sampling for inferring the parameters. Gibbs sampling cycles through the parameters, sampling each from its distribution conditioned on the current values of the other parameters. Because of the use of conjugate priors for the parameters, the posterior distributions are easy to sample from.

First, the posterior over parameter $\mathbf{U}$ is given. Each $\mathbf{U}_i$ is conditionally independent. The posterior distribution over $\mathbf{U}_i$ is given by:

$$p\left(\mathbf{U}_i|\mathbf{X}, \mathbf{V}, \alpha_X, \boldsymbol{\mu}_U, \mathbf{A}_U\right) = N\left(\mathbf{U}_i|\boldsymbol{\mu}^*_{U_i}, \mathbf{A}^*_{U_i}\right),$$

where

$$\boldsymbol{\mu}^*_{U_i} = \left[\mathbf{A}^*_{U_i}\right]^{-1}\left(\alpha_X \sum_{j=1}^{M} I^X_{ij}\left[x_{ij}\mathbf{V}_j\right] + \mathbf{A}_U\boldsymbol{\mu}_U\right)$$
$$\mathbf{A}^*_{U_i} = \mathbf{A}_U + \alpha_X \sum_{j=1}^{M} I^X_{ij}\left[\mathbf{V}_j\mathbf{V}_j^T\right].$$

Next, the posterior distribution of $\mathbf{V}_j$ is as follows:

$$p\left(\mathbf{V}_j|\mathbf{X}, \mathbf{F}, \mathbf{U}, \mathbf{V}_{-j}, \alpha_X, \alpha_F, \boldsymbol{\mu}_V, \mathbf{A}_V\right) = N\left(\mathbf{V}_j|\boldsymbol{\mu}^*_{V_j}, \mathbf{A}^*_{V_j}\right), \tag{4}$$

where

$$\boldsymbol{\mu}^*_{V_j} = \left[\mathbf{A}^*_{V_j}\right]^{-1} \left(\alpha_F \sum_{k=1,k\neq j}^{L} I^F_{jk} \left[f_{jk} \mathbf{V}_k \right] + \alpha_X \sum_{i=1}^{N} I^X_{ij} \left[x_{ij} \mathbf{U}_i \right] + \mathbf{A}_V \boldsymbol{\mu}_V \right)$$

$$\mathbf{A}^*_{V_j} = \mathbf{A}_V + \alpha_F \sum_{k=1,k\neq j}^{L} I^F_{jk} \left[\mathbf{V}_k \mathbf{V}_k^T \right] + \alpha_X \sum_{i=1}^{N} I^X_{ij} \left[\mathbf{U}_i \mathbf{U}_i^T \right].$$

Here $\mathbf{V}_{-j} = \{\mathbf{V}_k | k \neq j\}$ and I^X_{ij} is extended in such a way that it is equal to 0 if i or j exceeds the size of the matrix.

Next, we present the posterior distributions over the precisions of the data:

$$p\left(\alpha_X | \mathbf{X}, \mathbf{U}, \mathbf{V}, k_X, \theta_X\right) = Gam\left(\alpha_X | k^*_X, \theta^*_X\right),$$

where

$$k^*_X = k_X + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} I^X_{ij} \quad and \quad \theta^*_X = \theta_X + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} I^X_{ij} \left(x_{ij} - \mathbf{U}_i^T \mathbf{V}_j \right)^2,$$

and

$$p\left(\alpha_F | \mathbf{F}, \mathbf{V}, k_F, \theta_F\right) = Gam\left(\alpha_F | k^*_F, \theta^*_F\right),$$

where

$$k^*_F = k_F + \frac{1}{2} \sum_{j=1}^{L} \sum_{k=1}^{L} I^F_{jk} \quad and \quad \theta^*_F = \theta_F + \frac{1}{2} \sum_{j=1}^{L} \sum_{k=1}^{L} I^F_{jk} \left(f_{jk} - \mathbf{V}_j^T \mathbf{V}_k \right)^2.$$

That completes our presentation of the Gibbs sampling algorithm for CL-PMF.

4 Experiments

We examined the performance of CL-PMF, and compared it with the performance of existing dimension-reduction methods, by measuring the performance for the clustering of the compressed feature vectors. We assume that the better a dimension-reduction method compressed the feature vector, the better clusters are obtained with the compressed vectors. We used k-means clustering and used cosine similarity to measure the distance between two vectors. Since clustering performance depends on the number of clusters K, we ran k-means, varying the cluster number K.

In our experiments, we didn't directly cluster the tuples but instead defined the sequence of words on the dependency path from e_1 to e_2 as a relevant word sequence and clustered the relevant word sequences. We assume that the tuples sharing a relevant word sequence express a same relation. A feature vector of a relevant word sequence is the sum of the feature vectors of the tuples having the relevant word sequence.

Table 1. Descriptions of the datasets. N, M, and L are defined in Section 3.3, and nonzero is the average number of nonzero values in the feature vectors.

	SENT500	"Musicians"	"Companies"
N	328	2,159	861
M	297	4,754	4,548
L	7,750	24,955	21,852
nonzero	4.8	7.3	6.6

Table 2. Labels for "Musicians" and "Companies"

"Musicians"	"Companies"
collaboration	acquisition
appearance (TV)	location
tour (place)	part of
sign	partner
release	listing
member	production
reference	headquarter
win (award)	establishment
tour with	investment
birthplace	kind

4.1 Annotated Corpus

We performed clustering on three datasets: the published dataset SENT500 [2], a set of articles from the "Musicians" category in Wikipedia, and a set of articles from the "Companies" category in Wikipedia. Because there're few published dataset for our task, we manually built the second and the third dataset. The sizes of the datasets are shown in Table 1. Our target corpus is a small specific corpus in which almost all the entity pairs of tuples appear once, and each of the datasets from Wikipedia is such a corpus. In the "Musicians"/"Companies" dataset, the 98.6%/98.8% of the entity pairs appear once.

SENT500: The dataset consists of 500 sentences. Each sentence has an identified entity pair in one of the four relations: "acquisition", "birthplace", "inventor", "win (award)" [1]. The four relations are used as the gold standard. From SENT500 we obtained 328 relevant word sequences. Because we don't have an original corpus from which the sentences in SENT500 have extracted, we don't have enough sentences to gather feature co-occurrences. We used the sentences in the Wikipedia articles corresponding to the 50 entities identified in SENT500. (SENT500 includes the 50 unique entities and we used all of them.)

"Musicians" and **"Companies":** We applied the relation discovery method described in Section 3.1 to the Wikipedia articles in the "Musicians" category and the "Companies" category for evaluation in a realistic situation. In the case of the Wikipedia articles, one entity of a pair must be the title of an article, and the other needs to be an anchor text that links to another article and start with a capital letter. The relevant word sequences from both categories were labeled by hand for evaluation of clustering. For each dataset we prepared 10

[1] SENT500 actually has no labels of relations. We labeled the dataset with the four relations. The labels are available at
http://www.r.dl.itc.u-tokyo.ac.jp/~takamatsu/SENT500Label

labels of relations likely to have a large number of relevant word sequences in each corpus. We then labeled the relevant word sequences in one of the 10 labels. The labels differed between "Musicians" and "Companies." The used labels are shown Table 2. Because of the limited availability of space, we omit the definition of the labels. In the "Musicians"/"Companies" category we used the 4,997/4,063 articles and labeled 2,159/861 relevant word sequences.

4.2 Evaluation

We used as a measure of clustering performance the purity defined as follows:

$$purity = \frac{\sum_c N_c^{most}}{\sum_c N_c},$$

where N_c^{most} is the number of elements of the label that is most frequent in cluster c, N_c is the number of elements in cluster c, and the summation is over all the clusters. High purity in a small number of clusters indicates a good clustering performance. Our goal is to obtain a small number of clusters with high purity.

4.3 Methods

We used the following (compressed) feature vectors for input of k-means:

Raw Feature Vector (RFV): We used the raw feature vector $\mathbf{x}_i$ as a baseline and used the weighting function $W(x)$. The URE proposed in [5] employs this method.

Second-Order Feature Vector (2ndOrder): The feature vector was constructed by integrating the feature co-occurrence matrix in the feature vector $\mathbf{x}_i$ [12]. The feature vector $\mathbf{x'}_i = (x'_{ij}; j = 1, \ldots, L)$ is defined as $x'_{ij} = I^X_{ij} x_{ij} + \sum_{k=1}^{M} x_{ik} f_{kj}$. The feature vectors have information on the feature co-occurrence matrix $\mathbf{F}$. We applied $W(x)$ to each element of the feature vector $\mathbf{x'}_i$.

LSI: LSI inputs the feature vector matrix $\mathbf{X}$ and outputs the row vectors of the left singular matrix corresponding to the top D singular values. These row vectors are used as the compressed feature vectors.

PMF: We applied the standard PMF [9] to the feature vector matrix $\mathbf{X}$ and used the expectation of $\mathbf{U}_i$ over the posterior as the compressed feature vector.

CL-PMF: We chose the prior distributions for $\mathbf{U}$ and $\mathbf{V}$ with zero mean, $\boldsymbol{\mu}_U = \boldsymbol{\mu}_V = \mathbf{0}$, and precision matrix, $\mathbf{A}_U = \mathbf{A}_V = a\mathbf{I}$. The value of a determined by the experimental optimization was 15 for SENT500 and was 60 for "Musicians" and "Companies". We discuss the value of the precision a in Section 4.5. For the prior distributions of α_X and α_F we set $k_X = k_F = 1$ and $\theta_X = \theta_F = 1$.

The dimension-reduction methods described above input the feature vectors of the relevant word sequences including unlabeled ones, while we use labeled ones for clustering. We set D to 8 for SENT500, and to 20 for "Musicians" and "Companies". We discuss the value of D in Section 4.5.

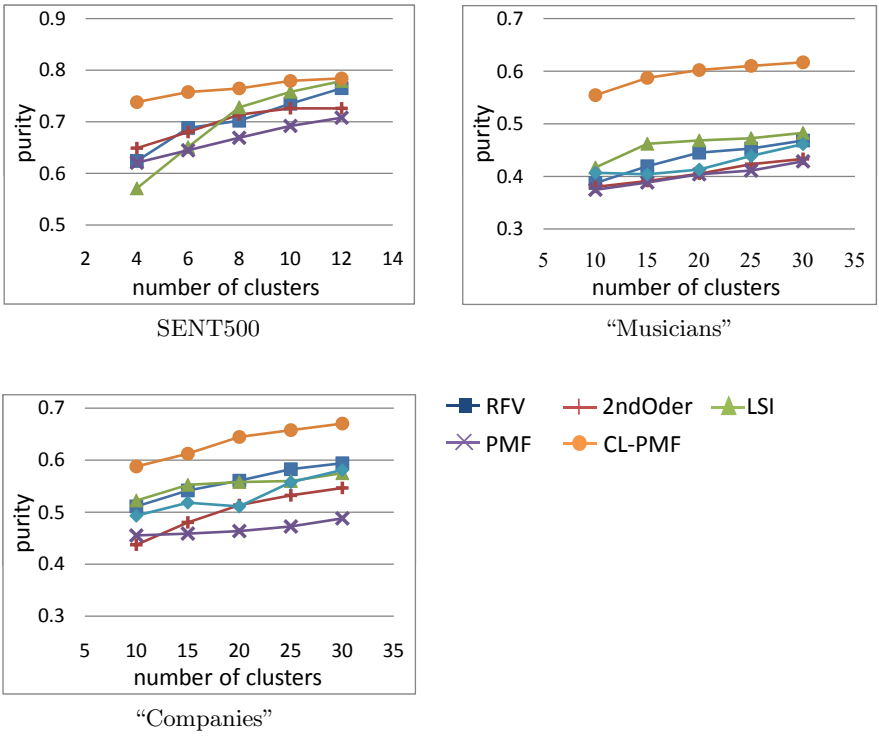


Fig. 2. Purities for the three data sets. The standard errors of the CL-PMF in the true K were 0.004 for SENT500, 0.015 for "Musicians" and 0.014 for "Companies."

4.4 Results and Discussion

In Fig. 2 the purities obtained with each of the methods are, for each dataset, plotted against the number of clusters. CL-PMF achieves the highest purity for each dataset and for each number of clusters. Note that CL-PMF outperforms others in a small number of clusters. The performances of LSI and PMF are inferior to or not far superior to those of RFV. This implies that a direct application of a standard dimension-reduction method doesn't work with the sparse feature vectors of the tuples, whereas CL-PMF works well because it uses the information of the feature co-occurrences in the entire corpus. CL-PMF performs better than 2ndOrder, which also utilizes the information of the feature co-occurrences. 2ndOrder is inferior to RFV because 2ndOrder is believed to be suffered from noise. CL-PMF leverages the feature co-occurrences better than 2ndOrder by introducing the compressed feature vectors $\mathbf{U}$ and assuming observation noise.

The purities of the CL-PMF are not relatively high in the large numbers of clusters in SENT500, where the clustering problem is so simple that RFV performs well. The feature co-occurrences are believed to have information for estimating the compressed feature vectors but also contain noise. In this simple

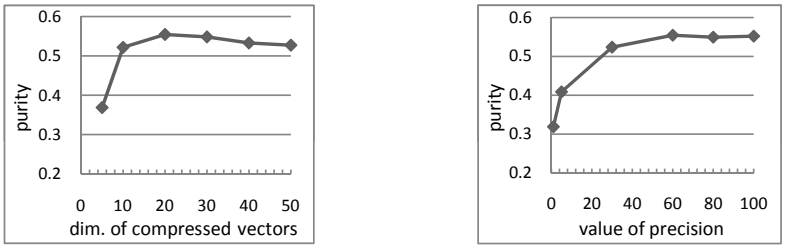


Fig. 3. The left panel shows purity as a fuction of D (when $a = 60$). The right panel shows purity as a function of a (when $D = 20$).

situation, the feature co-occurrences can become noise rather than a help for the estimation.

4.5 Parameters for CL-PMF

We performed experiments to see how the clustering was affected by the dimension D of the compressed feature vectors and the precision a of the prior distribution, using the "Musicians" dataset and setting K to 10.

The left panel in Fig. 3 shows that the purity peaks as the dimension D increases. The number of the dimension is believed to indicate the granularity of the semantic information. A small D prevents CL-PMF from capturing sufficient detail for accurate clustering. On the other hand, a large D causes overestimation, or excessive noise adaptation. The full Bayesian approach mitigates that.

The right panel in Fig. 3 shows a high value for the precision of the prior distributions, a, performs better than a low value. The prior distribution that has a high value of the precision keeps the compressed vectors $\mathbf{U}_i$ around the zero mean of the prior. This heavily takes into account their directions rather than their lengths when the compressed feature vectors are estimated. The similarities between the compressed vectors are measured well by the cosine similarity that we employed.

5 Conclusion

In this paper we have proposed CL-PMF for URE. CL-PMF compresses dimensions of sparse feature vectors of relational tuples, utilizing feature co-occurrences in the whole corpus including sentences where tuples are not extracted. Since our method doesn't assume the redundancy extracted from a huge amount of corpus, it works well for a small corpus such as that of a long tail domain. The experimental results show that the dimension reduction with CL-PMF is more effective for clustering of the sparse feature vectors of tuples than existing dimension-reduction methods and baseline methods.

References

1. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: Proceedings of IJCAI (2007)
2. Banko, M., Etzioni, O.: The tradeoffs between open and traditional relation. In: Proceedings of ACL 2008: HLT (2008)
3. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society of Information Science 41(6), 391–407 (1990)
4. Harris, Z.: Distributional structure. Word 10, 146–162 (1954)
5. Hasegawa, T., Sekine, S., Grishman, R.: Discovering relations among named entities from large corpora. In: Proceedings of ACL (2004)
6. Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: social recommendation using probabilistic matrix factorization. In: Proceedings of CIKM (2008)
7. Nakov, P., Popova, A., Mateev, P.: Weight functions impact on LSA performance. In: Proceedings of Euro Conference RANLP-2001 (2001)
8. Rosenfeld, B., Feldman, R.: URES: an unsupervised web relation extraction system. In: Proceedings of COLING/ACL (2006)
9. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Proceedings of NIPS (2008)
10. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In: Proceedings of ICML (2008)
11. Shuetze, H.: Dimensions of meaning. In: Proceedings of Supercomputing (1992)
12. Shuetze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–123 (1998)
13. Turney, P.D.: Measuring semantic similarity by latent relational analysis. In: Proceedings of IJCAI (2005)
14. Yates, A., Etzioni, O.: Unsupervised methods for determining object and relation systems on the web. Journal of Artificial Intelligence Research 34, 255–296 (2009)
15. Zhu, S., Yu, K., Chi, Y., Gong, Y.: Combining content and link for classification using matrix factorization. In: Proceedings of SIGIR (2007)

The Unsymmetrical-Style Co-training

Bin Wang[1], Harry Zhang[1], Bruce Spencer[2], and Yuanyuan Guo[1]

[1] Faculty of Computer Science, University of New Brunswick
P.O. Box 4400, Fredericton, NB, Canada E3B 5A3
[2] National Research Council of Canada,
Fredericton, NB, Canada E3B 9W4
bin.wang@unb.ca

Abstract. Semi-supervised learning has attracted much attention over the past decade because it provides the advantage of combining unlabeled data with labeled data to improve the learning capability of models. Co-training is a representative paradigm of semi-supervised learning methods. Typically, some co-training style algorithms, such as *co-training* and *co-EM*, learn two classifiers based on two views of the instance space. But they have to satisfy the assumptions that these two views are sufficient and conditionally independent given the class labels. Other co-training style algorithms, such as *multiple-learner*, use two different underlying classifiers based on only a single view of the instance space. However, they could not utilize the labeled data effectively, and suffer from the early convergence. After analyzing various co-training style algorithms, we have found that all of these algorithms have symmetrical framework structures that are related to their constraints. In this paper, we propose a novel unsymmetrical-style method, which we call the *unsymmetrical co-training algorithm*. The unsymmetrical co-training algorithm combines the advantages of other co-training style algorithms and overcomes their disadvantages. Within our unsymmetrical structure, we apply two unsymmetrical classifiers, namely, the self-training classifier and the EM classifier, and then train these two classifiers in an unsymmetrical way. The unsymmetrical co-training algorithm not only avoids the constraint of the conditional independence assumption, but also overcomes the flaws of the early convergence and the ineffective utilization of labeled data. We conduct experiments to compare the performances of these co-training style algorithms. From the experimental results, we can see that the unsymmetrical co-training algorithm outperforms other co-training algorithms.

1 Introduction

Over the course of the past decade, researchers have developed various types of semi-supervised learning methods. *Co-training* [1] is a representative paradigm of semi-supervised learning methods that are based on the multiple representations from difference views. Co-training was inspired by the observation discovered in the Web pages classification [1], in which a Web page has two different representations (views): the words occurring on the page itself; and the words

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 100–111, 2011.

contained in the anchor text of hyperlinks pointing to the page. The initial form of co-training is to train two classifiers separately on two sufficient and redundant views of data, and let these two classifiers label some unlabeled instances for each other. Like other semi-supervised learning methods, co-training requires its own assumptions to guarantee its success. Blum and Mitchell [1] proved that co-training can be successful if the two sufficient and redundant views are conditionally independent given the class label. Many researchers have supported the observation that co-training is sensitive to this theoretical assumption [18] [2]. However, the sufficient and redundant views are rarely found in most real-world application scenarios.

In order to tease out the effect of view-splitting from the effect of labeling, Nigam and Ghani [2] proposed a hybrid algorithm of expectation-maximization (EM) and co-training, called *co-EM*. Like co-training, co-EM tries to divide the instance space into two conditional independent views, and to train two EM classifiers based on these two views, respectively. But unlike co-training, co-EM uses all the unlabeled data every time, instead of incrementally selecting some confident predictions to update the training set. Nigam and Ghani [2] also provided a method for ideally splitting the view of instance space based on the conditional mutual information criteria between two subsets of attributes. However, this method is NP-hard and difficult to apply in practice.

Since both co-training and co-EM suffer from the conditional independence assumption, variants of co-training have been developed based on only a single view (without splitting the attribute set). For example, Goldman and Zhou [3] used two different learning algorithms in the paradigm of co-training without splitting the attribute set. Steedman et al. [4] developed a similar co-training algorithm that applies two diverse statistical parsers. Wang and Zhou [5] proved that if the two classifiers are largely diverse, co-training style algorithms are able to succeed. Because these variants substitute multiple views by multiple classifiers, these algorithms are referred to as *multiple-learner* algorithms. Since the multiple-learner algorithms are trained on the same attribute set, it is important to keep the two classifiers different during the process in order to prevent early convergence. Maintaining separated training sets is one approach for this purpose. However, assigning labeled instances to two different initial training sets will cause the ineffective utilization of labeled data sets in a semi-supervised learning scenario.

Considering the framework structures of co-training, co-EM, and multiple-learner algorithms, we can see that each structure is symmetrical. The co-training algorithm splits the instance space into two symmetrical views, trains two classifiers symmetrically, and lets two classifiers teach each other in a symmetrical way. Similarly, the co-EM algorithm sets up two symmetrical EM classifiers based on their related views. And likewise, the multiple-learner algorithm also has a symmetrical structure, where two classifiers are trained in parallel and combined together to score the unlabeled instances. Therefore, we define these algorithms as the *symmetrical-style co-training algorithms*.

In this paper, we propose an *unsymmetrical co-training algorithm* - a novel, semi-supervised, unsymmetrical-style algorithm. The unsymmetrical co-training algorithm combines the advantages of other co-training style algorithms and overcomes their disadvantages. The unsymmetrical co-training algorithm combines two unsymmetrical classifiers, namely, an EM classifier and a self-training classifier. In the algorithm, the self-training classifier takes the responsibility for a section of unlabeled instances in a data pool; and the EM classifier maintains a global view over the entire instance set. Without the two-view splitting, the unsymmetrical co-training algorithm uses the full set of attributes so that it can avoid the intractable constraint of the conditional independence assumption. Although both classifiers are initially trained by labeled instances, the EM classifier and the self-training classifier have different training sets once they enter the iteration procedure. The unsymmetrical co-training algorithm does not need to hold different initial training sets and is therefore able to utilize the labeled instances more effectively. Furthermore, according to the study of Wang and Zhou [5], the multiple-learner algorithm could not further improve the performance after a number of learning rounds because the difference between the two learners become smaller and smaller. Since the unsymmetrical co-training algorithm uses two unsymmetrical classifiers in an unsymmetrical structure, it does not need to worry about the difference between these two classifiers fading too quickly. We conduct the experiments to compare the performances of co-training, co-EM, multiple-learner, and unsymmetrical co-training algorithms on 30 data sets from Weka [6]. From the experimental results, we can see that the unsymmetrical co-training algorithm outperforms other algorithms.

The remainder of this paper is organized as follows. After introducing some preliminaries about co-training, co-EM, and multiple-learner algorithm in Section 2, we present our unsymmetrical co-training algorithm in Section 3. Then, the experiments to compare the performances of algorithms are reported in Section 4. Finally, we give our conclusion and look toward future work in Section 5.

2 Preliminaries

Suppose we have the instance space $X = X^1 \times X^2$, where X^1 and X^2 correspond to the two different views of the instance space, respectively, and the class label space Y. Given the data set $L \cup U$, we have a labeled data set $L = \{\langle (x_1^1, x_1^2), y_1 \rangle, \cdots, \langle (x_l^1, x_l^2), y_l \rangle\} \subset X \times Y$ and an unlabeled data set $U = \{(x_{l+1}^1, x_{l+1}^2), \cdots, (x_{l+u}^1, x_{l+u}^2)\} \subset X$. In addition, we have two classifiers h_1 and h_2, which are used to compose the following algorithms.

2.1 Co-training

Co-training [1] first tries to divide the instance space X into two different views X_1 and X_2, which are conditionally independent given the class label. Then, two classifiers h_1 and h_2 are trained based on these two different views, respectively. Classifier h_1 classifies the unlabeled instances and "teaches" the other classifier h_2 the predicted class labels of unlabeled instances about which it feels most

confident. These confident unlabeled instances are added into the training set of h_2 together with their predicted class labels. At the same time, classifier h_2 teaches h_1 the predicted class labels about which it feels more confident. After that, each classifier is retrained with the updated training set. Such a process can be repeated until a certain stopping condition is satisfied. The framework structure of co-training is shown in Figure 1. From Figure 1, we observe that the framework structure of co-training is symmetrical: the instance space is divided into two views symmetrically; the two classifiers are trained symmetrically; and they update each other's training set in a symmetrical way.

Some theoretical studies have analyzed why and how the co-training algorithm can succeed. With proposing the co-training algorithm, Blum and Mitchell [1] defined the co-training model in a PAC-style theoretical framework and proved that the two different views are supposed to satisfy the following conditions: (1) each view is sufficient and consistent to train a good classifier; (2) each view is conditionally independent to the other one given the class label. Dasgupta et al. [20] also provided a PAC-style theoretical analysis for co-training. Yu et al. [7] proposed a graphical model for the co-training algorithm based on the conditional independence assumption. Abney [8] showed that weak dependence can also guarantee co-training's working. Balcan et al. [9] proposed a much weaker "expansion" assumption on the underlying data distribution, and proved that it is sufficient for the iterative co-training to succeed. Wang and Zhou [10] analyzed the co-training algorithm as a combinative label propagation over two views, and provided the sufficient and necessary condition for co-training to succeed.

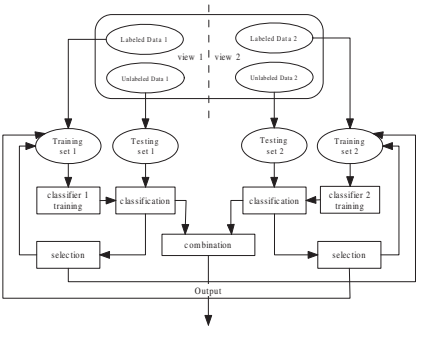


Fig. 1. The structure of co-training algorithm

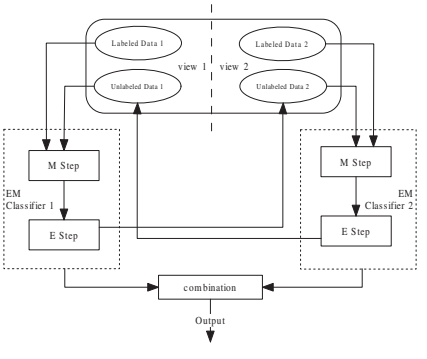


Fig. 2. The structure of co-EM algorithm

2.2 Co-EM

Nigam and Ghani [2] compared the performances of the co-training algorithm with the EM algorithm, and studied how sensitive the co-training algorithm is to the conditional independence assumption. They the proposed a hybrid algorithm of EM and co-training, called co-EM. The co-EM algorithm is similar to the EM algorithm, which is an iterative procedure that uses all the unlabeled instances every

time, instead of incrementally selecting some unlabeled instances to update the training set. On the other hand, the co-EM algorithm is also like the co-training algorithm, which tries to divide the instance space into two conditionally independent views. Nigam and Ghani [2] argued that the co-EM algorithm is a closer match to the theoretical argument established by Blum and Mitchell [1].

In the co-EM algorithm, two EM classifiers, h_1 and h_2, are chosen to correspond to the two different views X_1 and X_2. Initially, classifier h_1 is trained only based on the labeled data set L with view X_1. Then, classifier h_1 probabilistically labels all the unlabeled instances in data set U. Next, classifier h_2 is trained using the labeled instances from the view X_2 of data set L, plus the unlabeled instances from the view X_2 of data set U with the class labels given by h_1. Classifier h_2 then relabels the instances for the retraining of h_1. This process iterates until the classifiers converge. The framework structure of co-EM is shown in Figure 2. From Figure 2, we can see the symmetrical structure of the co-EM algorithm.

2.3 Multiple-Learner

Since the paradigmatic assumptions of co-training are difficult to satisfy in real-world application scenarios, many researchers begin to study the variants of co-training that do not require the two-view splitting [3] [4] [11]. Because those algorithms usually use multiple learners, they are referred to as the multiple-learner algorithms. Ng and Cardie [12] summarized multiple-learner algorithms and proposed their own algorithm[1]. In their multiple-learner algorithm, two different classifiers h_1 and h_2 are used and trained based on the single view of instance space. At each iteration, each classifier labels and scores all the instances in a data pool. Some instances with scores found to be high by classifier h_1 are added to the training set of classifier h_2 together with their predicted class labels from h_1, and vice verse. Then, the entire data pool is flushed and replenished using instances drawn from the unlabeled data set U after each iteration. The process is repeated until no further instances can be labeled. The framework structure of multiple-learner algorithm is shown in Figure 3. Here we can see that the structure of multiple-learner algorithm is also symmetrical, where two classifiers are trained in parallel and combined together to score the unlabeled instances in the data pool.

3 Unsymmetrical Co-training

As we have already emphasized, the co-training algorithm, the co-EM algorithm, and the multiple-learner algorithm all have symmetrical framework structures. Some of the constraints that restrict these algorithm are related to their symmetrical structures. For example, both the co-training algorithm and the co-EM

[1] The multiple-learner algorithm that appears in later sections refers to the version of Ng and Cardie [12]

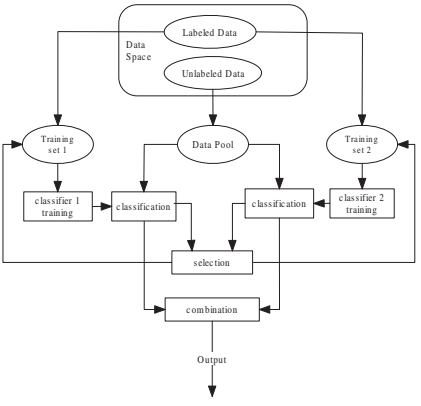


Fig. 3. The structure of multiple-learner algorithm

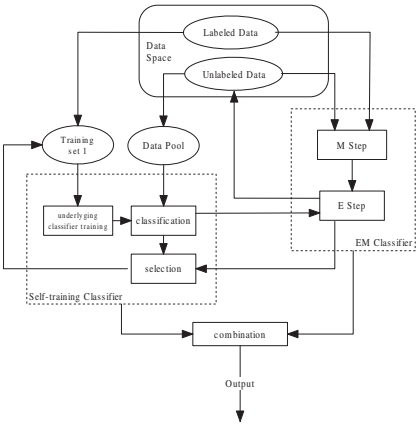


Fig. 4. The structure of unsymmetrical co-training algorithm

algorithm are required to symmetrically divide the instance space into two sufficient and conditionally independent views, and their performances are quite sensitive to these assumptions. However, fulfilling these assumptions is a NP-hard problem, and these assumptions are intractable in practice. Although the multiple-learner algorithm does not suffer from the same intractable assumptions, it nevertheless still requires two different classifiers that are trained in a symmetrical way. Wang an Zhou [10] reported that if the two initial classifiers have large difference, they can be improved together using the multiple-learner algorithm. They also discovered that, as the multiple-learner algorithm proceeds, more and more unlabeled data are labeled, which makes the difference between the two learners become smaller and smaller. In other words, even though the two classifiers have big difference initially, they become more and more similar after several learning rounds since they are trained in a symmetrical way, and the performance of the multiple-learner algorithm cannot be further improved. Moreover, if the two selected classifiers are only slightly different from one another, two different training sets are required in order to avoid convergence of the algorithm at an early stage in the symmetrical structure. However, in the scenario of semi-supervised learning, assigning labeled instances to two different initial training sets will cause the ineffective utilization of labeled data set. To escape the constraints of symmetrical structures, we attempt to design a new algorithm that still performs with the style of co-training but has an unsymmetrical framework structure.

In this paper, we propose the unsymmetrical co-training algorithm. This algorithm uses two unsymmetrical classifiers, namely, the EM classifier and the self-training classifier. The EM classifier is a kind of generative model that has been successfully applied in semi-supervised learning [13]. The EM algorithm includes two steps: the E-step and the M-step. The E-step estimates the expectations of the class information of unlabeled instances, and the M-step maximizes

the likelihood of the model parameters using the expectations from the previous E-step. The EM classifier performs an iterative hill-climbing process to find a local maxima of model probability and then assigns the class labels in terms of the established model. Self-training might be the earliest technique for semi-supervised learning [14] [15] and is commonly applied in many domains [17] [16]. In the self-training classifier, an underlying classifier is iteratively trained and used to label the unlabeled instances, and then some unlabeled instances having the confident predictions are used to update the training set for the next round training of the underlying classifier.

Not only are two unsymmetrical classifiers applied in the unsymmetrical co-training algorithm, but these two classifiers are also applied within an unsymmetrical framework structure. In the structure, a data pool has been created for the self-training classifier by randomly selecting instances from the unlabeled instance set U. This data pool is the labeling objective on which the self-training classifier focuses in the process of the algorithm. But the EM classifier does not focus on a specific section of unlabeled instances. Instead, it faces the entire unlabeled instance set U during the algorithm. Moreover, the training sets used to train these two classifiers are different. For the self-training classifier, the training set consists of the labeled instances and the unlabeled instances from the data pool with their predicted class labels. For the EM classifier, the training set is the labeled instances plus the whole unlabeled instances together with the class labels assigned from the previous learning round. The framework structure of unsymmetrical co-training algorithm is shown in Figure 4.

The unsymmetrical co-training algorithm learns the two classifiers in an unsymmetrical way. Initially, both of the classifiers are trained based on the labeled instance set L with the single view (the full set of attributes). Then, the EM classifier labels all the unlabeled instances, and the self-training classifier predicts the labels of unlabeled instances in the data pool. The predicted class labels of unlabeled instances in the data pool will be used to substitute the class labels of corresponding unlabeled instances that have been assigned by the EM classifier. The EM classifier is then retrained by the updated training set and relabels the unlabeled instances. The unlabeled instances in the data pool, for which class labels from the self-training classifier are identical to the class labels from EM classifier, will be selected to update the training set of self-training classifier together with their predicted class labels. If there are not enough such unlabeled instances, the confidence degree metric will be used to select other unlabeled instances with high confidence in the data pool to update the training set together with the predicted class labels from the self-training classifier. Next, the data pool will be replenished by other unlabeled instances. The procedure is repeated until there are no further instances in the data pool. The predictions for new-coming instances are obtained using the combination of predictions from the EM classifier and the self-training classifier, just as the co-training algorithm does [1]. The formal description of unsymmetrical co-training algorithm is shown in Figure 5.

According to the theoretical study of Wang and Zhou [5], the co-training style algorithm is able to succeed if the two classifiers are vastly different. From the

Input: Labeled instance set L, unlabeled instance set U, self-training classifier h_{self}, and EM classifier h_{EM}.
Initialization:

- Initialize the training set for h_{self} by L.
- Create the data pool for h_{self} by randomly selecting from U.
- Build h_{EM} by L, and use h_{EM} to label all the unlabeled instances in U.
- Create the training set for h_{EM} using all the labeled instances and unlabeled instances with predicted class labels from h_{EM}.

Loop if h_{self}'s data pool still has some instances

- Build h_{self} using its training set.
- Use h_{self} to label the instances in its data pool.
- Use the predicted labels in h_{self}'s data pool to replace the corresponding instances' class labels in h_{EM}'s training set.
- Build h_{EM} by the updated training set.
- Relabel all the unlabeled examples using h_{EM}, and return the predicted class labels of instances in the data pool to h_{self}.
- h_{self} selects some instances in the data pool if:
  1. their predicted labels are identical to the labels from h_{EM}; or
  2. they have higher scores ranked by the confidence degree metric of h_{self}.
- Add selected instances into the training set of h_{self} together with their predicted class labels.
- Replenish instances in the data pool using other unlabeled instances.

Output: The combination of predictions for new-coming instances from h_{self} and h_{EM}.

Fig. 5. The description of unsymmetrical co-training algorithm

description above, we can see that the unsymmetrical co-training is consistent with their theory. The self-training classifier and the EM classifier not only display different characteristics on learning, but also are deployed differently in the unsymmetrical framework structure. Although they are both initially trained by the same labeled instances, these two classifiers have different training sets once they enter the iteration procedure. Therefore, the unsymmetrical co-training algorithm avoids the early convergence of both classifiers to the same hypothesis, and utilizes the labeled and unlabeled instances more effectively than does the multiple-learner algorithm. Moreover, unlike the multiple-learner algorithm, in which two classifiers become more and more similar as the algorithm proceeds, our algorithm always maintains the differences between the two classifiers due to its unsymmetrical way of learning. On the other hand, the unsymmetrical co-training algorithm uses the single view of instance space so that it avoids the intractable conditional independent view-splitting.

In the unsymmetrical co-training algorithm, the self-training classifier and the EM classifier can complement each other. The EM algorithm essentially uses the naive Bayes method to assign class membership probabilities to unlabeled instances. The EM classifier is expected to do well when it satisfies the conditional independence assumption of naive Bayes. However, these probabilities are always poorly estimated because the conditional independence assumption is violated. Self-training, on the other hand, uses the class membership probabilities for ranking the confidences of unlabeled instances instead of directly using them for the classification. Thus, the conditional independence assumption influences the self-training classifier more weakly than does the EM classifier. From another point of view, self-training is an incremental procedure and always suffers from the reinforcement of any misclassifications from previous updates. In the unsymmetrical co-training algorithm, the EM classifier is like a supervisor beside the self-training classifier, which checks the predicted class labels made

by self-training and provides opinions for these predictions in terms of its own knowledge. It is useful to reduce the chance of adding misclassifications to the next iteration in the procedure. Moreover, the self-training classifier restricts its view only on the labeled instances and the unlabeled instances in the data pool. Since the EM classifier is able to see the entire set of labeled and unlabeled instances, the view of the EM classifier is much broader than that of the self-training classifier. Therefore, the EM classifier might be seen to make predictions from a global view to help self-training, and the self-training classifier likewise boosts EM from its local view.

We summarize the differences of co-training, co-EM, multiple-learner, and unsymmetrical co-training algorithms from several points of view in Table 1. From Table 1, we can see that the unsymmetrical co-training algorithm not only combines the advantages of other algorithms, but also overcomes their disadvantages. In the next section, we design the experiments to compare the performances of these algorithms in the next section.

Table 1. The differences of co-training, co-EM, multiple-learner and unsymmetrical co-training algorithms

	co-training	co-EM	mult-learner	unsym co-training
Structure	Symmetrical	Symmetrical	Symmetrical	Unsymmetrical
Split attribute set	Yes	Yes	No	No
Split training set	No	No	Yes	No
Learning style	Incremental	Iterative	Incremental	Incremental and iterative
Num of instances added per iteration	Fixed	Variable	Fixed	Variable
Use data pool	Yes	No	Yes	Yes
Example selection for two learners	Higher scored	N/A (no need to select)	Agreed and higher scored	Agreed and higher scored

4 Experiments

In this section, we design the experiments to compare the performances of co-training, co-EM, multiple-learner, and unsymmetrical co-training algorithms. The experiments are conducted on 30 data sets from Weka [6], which are selected from the UCI repository. There are some preprocessing stages adopted on each data set. First, we use the filter $ReplaceMissingValues$ in Weka to replace the missing values of attributes in each data set. Second, we use the filter $Discretize$ in Weka, which is the unsupervised ten-bin discretization, to discretize numeric attributes. Thus, all the attributes are nominal. Moreover, we notice that some attributes do not contribute any information for the purpose of prediction if the numbers of these attributes are almost equal to the numbers of instances in the corresponding data sets. The third preprocessing stage is to use the filter *Remove* in Weka to delete such attributes. We implement co-training, co-EM, multiple-learner and unsymmetrical co-training algorithms in the Weka framework. The underlying classifiers used by algorithms are naive Bayes classifiers, except the co-EM and a part of the unsymmetrical co-training algorithms that use the EM classifiers.

Our experiments are configured as follows. In each data set, 10% of the instances are used as testing instances; 10% of the remaining data set is used as the

Table 2. Experimental results on accuracy for co-training, co-EM, multiple-learner and unsymmetrical co-training algorithms

Dataset	UnSymCoTrain	Co-training	Co-EM	Mult-Learner
zoo	85.36	79.48	76.24	80.19
labor	78.83	78.5	65.9	79.87
iris	89.2	89.33	79.47	90.4
vote	88.62	88	88.23	88.26
breast-cancer	72.25	65.94	71.63	70.97
lymph	74.21	55.1	64.65	57.63
primary-tumor	34.29	30.98	25.38	30.44
hepatitis	82.55	80.5	80.08	82.18
balance-scale	72.38	70.03	53.68	65.63
glass	45.43	42.27	42.42	43.53
audiology	31.46	34.27	26.55	35.16
heart-h	83.2	74.27	81.36	79.12
heart-c	81.99	67.27	82.75	74.53
colic.ORIG	58.45	59.02	54.88	57.38
heart-statlog	81.04	77.67	70.07	81.07
autos	46.79	45.58	41.22	45.67
credit-a	84.29	82.14	75.64	83.88
colic	72.2	74.17	67.37	73.54
breast-w	97.44	97.02	97.4	97.08
diabetes	71.66	71.79	69.37	70.75
anneal.ORIG	77.04	77.38	69.32	77.61
soybean	77.71	68.3	62.29	70.77
ionosphere	84.64	80.34	80.6	82.73
anneal	86.15	83.44	68.93	84.66
vowel	30.59	26.82	18	26.86
kr-vs-kp	65.55	74.27	51.63	72.82
credit-g	69.03	67.51	67.13	65.73
vehicle	50.33	47.26	43.91	46.34
sonar	65.23	56.55	55.7	56.82
mushroom	90.04	93.06	89.24	92.96
w/t/l		2/20/8	0/20/10	2/20/8

set of labeled instances; and all other instances are used as unlabeled instances. For the co-training and co-EM algorithms based on two views, the attribute set is randomly divided into two disjointed subsets. For the co-training, co-EM, and unsymmetrical co-training algorithms that need the data pool, the size of the data pool is set to 10% of the unlabeled instance set. For the co-training and multiple-learner algorithms that fix the number of instances added per iteration, the number of added instances for each class label is decided by the class label distribution in the original labeled instance set: for the class label with the minimum percentage, the number is set to 1; and for all the other class labels, the numbers are set to the times of that the class label has the minimum percentage. The accuracy score is used to evaluate the performances of algorithms. In our experiments, the accuracy scores of each algorithm are obtained via 10 runs of ten-fold cross-validation and evaluated on the same testing sets. Finally, we conduct two-tailed t-test with a 95% confidence level to compare the unsymmetrical co-training algorithm to the other algorithms. The results are shown in Table 2.

In Table 2, the two-tailed t-test results are shown in the bottom row, where each entry has the format of $w/t/l$. This means that, comparing with the unsymmetrical co-training algorithm, the algorithm in the corresponding column wins w times, ties t times, and loses l times. From the experimental results, we observe that the unsymmetrical co-training algorithm outperforms other algorithms, where it wins 8 times and loses 2 times against the co-training and multiple-learner algorithms, and wins 10 times and never loses against the co-EM algorithm.

The evidences provided by the above experiments can be explained as follows. The unsymmetrical structure is more effective in the scenario of semi-supervised learning than are the symmetrical structures. The unsymmetrical co-training

algorithm combines the EM classifier, which learns from a global view, and the self-training classifier, which boosts the learning from the local view. These two classifiers, which complement each other in the unsymmetrical structure, enhance the learning ability of the co-training style framework. The co-training and co-EM algorithms are sensitive to the conditional independence assumption. Randomly splitting the attribute sets decreases the overall performances of co-training and co-EM algorithms. Although the multiple-learner algorithm does not need two-view splitting, just as the unsymmetrical co-training algorithm does, the small number of labeled instances cannot be utilized effectively in the symmetrical structure. Besides, as the multiple-learner algorithm proceeds, its underlying classifiers become more and more similar so that the performance cannot be further improved.

5 Conclusion

In this paper, we propose the unsymmetrical co-training algorithm, which is a novel semi-supervised learning method in the co-training style. In the algorithm, two unsymmetrical classifiers, namely, the self-training classifier and the EM classifier, are learned in an unsymmetrical way within an unsymmetrical framework structure. Compared with other co-training style algorithms, such as co-training, co-EM, and multiple-learner, the unsymmetrical co-training algorithm has several advantages. First, the unsymmetrical co-training algorithm is based on the single view of instance space, so it does not suffer from the violation of conditional independence assumption as co-training and co-EM algorithms do. Second, the unsymmetrical structure makes the utilization of labeled instances more effective since there is no need to split the labeled instance set into two different initial training sets for two underlying classifiers. Moreover, the two unsymmetrical classifiers do not easily become more similar after several learning rounds because the unsymmetrical training of the algorithm prevents growing similarity. We conduct the experiments to compare the performances of these algorithms. The experimental results show that the unsymmetrical co-training algorithm overall outperforms other algorithms.

In the future, we will continue the study of semi-supervised learning methods in the co-training style, especially within the unsymmetrical framework structure. More experiments will be conducted to compare our unsymmetrical co-training algorithm with other co-training style methods under various circumstances. The different underlying classifiers will be tested within this unsymmetrical structure to see whether the performance can be improved further. It will also be interesting to apply the unsymmetrical co-training algorithm to real-world applications, especially for the applications suitable for semi-supervised learning, such as natural Language processing (NLP) and bioinformatics.

References

1. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 1998 Conference on Computational Learning Theory, pp. 92–100. Morgan Kaufmann, San Francisco (1998)

2. Nigam, K., Ghani, R.: Analyzing the Effectiveness and Applicability of Co-training. In: CIKM, pp. 86–93 (2000)
3. Goldman, S.A., Zhou, Y.: Enhancing Supervised Learning with Unlabeled Data. In: Proceedings of the Seventeenth International Conference on Machine Learning, pp. 327–334 (2000)
4. Steedman, M., Osborne, M., Sarkar, A., Clark, S., Hwa, R., Hockenmaier, J., Ruhlen, P., Baker, S., Crim, J.: Bootstrapping statistical parsers from small datasets. In: Proceedings of the EACL, pp. 331–338 (2003)
5. Wang, W., Zhou, Z.-H.: Analyzing Co-training Style Algorithms. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 454–465. Springer, Heidelberg (2007)
6. Witten Ian, H., Frank, E.: Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco (2002)
7. Yu, S., Krishnapuram, B., Rosales, R., Steck, H., Rao, R.B.: Bayesian co-training. NIPS 20, 1665–1672 (2008)
8. Abney, S.: Bootstrapping. In: ACL, pp. 360–367 (2002)
9. Balcan, M., Blum, A., Yang, K.: Co-Training and Expansion: Towards Bridging Theory and Practice. In: Advances in Neural Information Processing Systems, vol. 17, pp. 89–96 (2004)
10. Wang, W., Zhou, Z.H.: A new analysis on co-training. In: Proceedings of the 27th International Conference on Machine Learning (2010)
11. Zhou, Z.-H., Li, M.: Semi-supervised regression with co-training. In: IJCAI 2005: Proceedings of the 19th International Joint Conference on Artificial Intelligence, San Francisco, CA, USA, pp. 908–913 (2005)
12. Ng, V., Cardie, C.: Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms. In: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 113–120 (2003)
13. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning 39, 103–134 (2000)
14. Hearst, M.: Noun homograph disambiguation using local context in large text corpora, pp. 1–22. University of Waterloo, Waterloo (1991)
15. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, pp. 189–196 (1995)
16. Wang, B.: Semi-supervised Learning and Opinion-oriented Information Extraction. PhD dissertation, University of New Brunswick (2010)
17. Wang, B., Spencer, B., Ling, C.X., Zhang, H.: Semi-supervised self-training for sentence subjectivity classification. In: Bergler, S. (ed.) Canadian AI. LNCS (LNAI), vol. 5032, pp. 344–355. Springer, Heidelberg (2008)
18. Muslea, I., Minton, S., Knoblock, C.: Active + Semi-Supervised Learning = Robust Multi-View Learning. In: Proceedings of the 19th International Conference on Machine Learning (ICML 2002), pp. 435–442 (2002)
19. Ganchev, K., Graca, J.V., Blitzer, J., Taskar, B.: Multi-view learning over structured and non-identical outputs. In: Proceedings of the Converence on Uncertainty in Artificial Intelligence (UAI) (2008)
20. Dasgupta, S., Littman, M.L., Mcallester, D.: Pac generalization bounds for co-training. In: NIPS (2001)

Balance Support Vector Machines Locally Using the Structural Similarity Kernel

Jianxin Wu

School of Computer Engineering, Nanyang Technological University

Abstract. A structural similarity kernel is presented in this paper for SVM learning, especially for learning with imbalanced datasets. Kernels in SVM are usually pairwise, comparing the similarity of two examples only using their feature vectors. By building a neighborhood graph (kNN graph) using the training examples, we propose to utilize the similarity of linking structures of two nodes as an additional similarity measure. The structural similarity measure is proven to form a positive definite kernel and is shown to be equivalent to a regularization term that encourages balanced weights in all local neighborhoods. Analogous to the unsupervised HITS algorithm, the structural similarity kernel turns hub scores into signed authority scores, and is particularly effective in dealing with imbalanced learning problems. Experimental results on several benchmark datasets show that structural similarity can help the linear and the histogram intersection kernel to match or surpass the performance of the RBF kernel in SVM learning, and can significantly improve imbalanced learning results.

1 Introduction

Measuring the degree of similarity between two examples is a critical task in machine learning. Many positive definite kernels used in the support vector machines (SVM) can be considered as similarity measures, such as the linear kernel $k_{\text{LIN}}(x_1, x_2) = x_1^T x_2$, the RBF kernel $k_{\text{RBF}}(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2)$, or the histogram intersection kernel [14]

$$k_{\text{HI}}(x_1, x_2) = \sum\nolimits_j \min(x_{1j}, x_{2j}) \; . \tag{1}$$

However, these kernels which only take into account the two features vectors x_1 and x_2 might provide misleading similarity scores, especially when the intra-class variation is large. Fig. 1 illustrates this idea. In the middle row of Fig. 1, the digit '7' (example x_2) is the nearest neighbor of the digit '2' (example x_1) when the RBF kernel is used.

A kernel that only compares pairwise similarities is susceptible to such undesired mismatches. However, Fig. 1 also illustrates that the structural similarity can greatly help remove the ambiguities. We use the notation $x_1 \mapsto x_2$ to indicate that x_2 is among the nearest neighbors of x_1. The first row in Fig. 1 shows the other ten nearest neighbors of x_1 ('2') excluding x_2 ('7') in the decreasing

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 112–123, 2011.

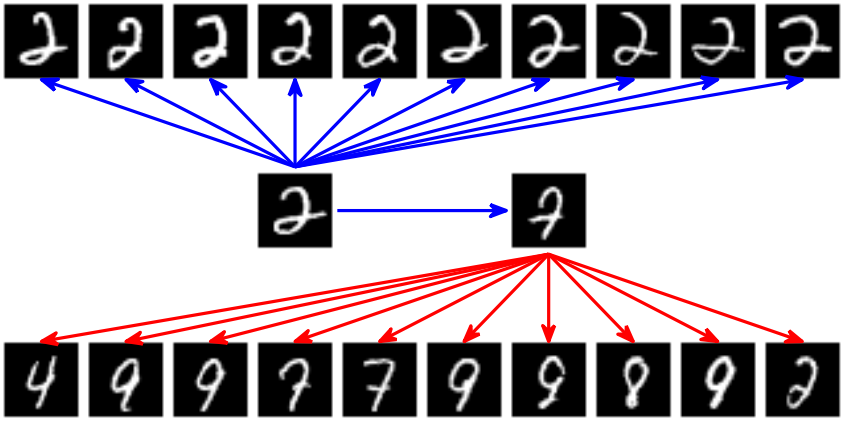


Fig. 1. Structural similarity helps disambiguate the digits '2' and '7' in the middle row

order of similarity. And the third row shows the nearest neighbors of x_2 (x_1 is not in the neighborhood of x_2 in this example). Since x_1 and x_2 do not share any common neighbor (i.e., x_1 and x_2 are exhibiting completely different linking structures in the neighborhood graph), they are not similar in terms of the structural similarity. A combination of the pairwise and structural similarity will generally improve the classification of examples.

In this paper we propose to use bibliographic coupling strength [6] to measure the structural similarity of examples x_1 and x_2, i.e., counting the number of examples that are nearest neighbors of both x_1 and x_2. One advantage of this simple structural similarity measure is that we can encode a local graph as a sparse neighborhood vector. By appending the neighborhood vector to the original feature vector, existing SVM training and testing techniques can be performed without any change if the linear or histogram intersection kernel is used. The similarity of two examples, though, becomes a combination of both the pairwise similarity and the structural similarity.

We further prove that when the linear or histogram intersection kernel is used, graph structural similarity is equivalent to adding a regularization term that encourages *local balance*. Given a set of labeled examples $(x_i, y_i), i = 1, \ldots, n$, $y_i \in \{-1, +1\}$, a constraint $\sum_i \alpha_i y_i = 0$ is enforced for dual variables α_i in the dual SVM problem if a bias term is used. We will call this constraint the global balance constraint. Instead, structural similarity leads to the minimization of $\|s\|^2$, where $s = [s_1, \ldots, s_n]^T$ and $s_j = \sum_{i:x_i \mapsto x_j} \alpha_i y_i$. We call s_j the local balance term because it measures the balance of α in the small subset of examples that emit edges toward x_j. SVM classifiers usually suffer from imbalanced data distributions. When one class (the minority class) has much less examples than another class (the majority class), the minority class accuracy is generally significantly lower than that of the majority class. SVM with structural similarity leads to similar performances in both the minor and the major classes because of the locally balanced property. We also show that the local balance term s is a

natural supervised learning extension of the authority scores in the unsupervised HITS algorithm [9].

Experimented on 6 benchmark datasets, the proposed locally balanced SVM with structural similarity shows improved accuracies in most datasets than normal SVM. In imbalanced problems, the proposed method can increase the minority class accuracy by up to 100%, *without harming the majority class accuracy.*

The rest of this paper introduces the structural similarity kernel (Sec. 2), the local balance property (Sec. 3), and the signed authority score interpretation (Sec. 4). After discussing related research in Sec. 5, experiments (with discussions of limitations of the proposed method) are presented in Sec. 6.

2 Bibliographic Coupling Based Structural Similarity

Bibliographic coupling was originally proposed for comparing scientific documents [6]. For documents p and q, it is defined as the number of documents cited by both p and q, based on the intuition that p and q are closely related if they share many common bibliography items. A similar intuition applies to structural similarity: if the set of nearest neighbors of x_1 and x_2 share many common members, then it is highly probable that x_1 and x_2 are in the same class.

The nearest neighbor relationship, however, depends on a training set $X = \{x_1, \ldots, x_n\}$. As illustrated in Fig. 1, a kNN graph Γ_X can be built for X, where a data point x_i corresponds to a node in Γ_X, and a directed edge from x_i to x_j is created if and only if x_j is in the k-nearest-neighbors of x_i (denoted as $x_i \mapsto x_j$).[1] We then define the structural similarity kernel as

$$k_{\mathrm{STR}}(x_i, x_j) = \left|\{x : x \in X, x_i \mapsto x \text{ and } x_j \mapsto x\}\right| , \tag{2}$$

in which $|\cdot|$ is size of a set. Note that k_{STR} is a data-dependent kernel – it is defined with respect to the training set X. In order to use k_{STR} in an SVM, we need to prove that it is a positive definite kernel. Let G be the adjacency matrix of Γ_X, i.e., G_{ij} equals 1 if $x_i \mapsto x_j$, and 0 if otherwise. It is easy to see that $K^{\mathrm{STR}} = GG^T$, where K^{STR} is the kernel matrix satisfying $K^{\mathrm{STR}}_{ij} = k_{\mathrm{STR}}(x_i, x_j)$.

Alternatively, we can define a neighborhood vector $n(x)$ for any sample x. $n(x)$ is defined as a $n \times 1$ vector where $n(x)_j$ equals 1 if x_j is in the k-nearest-neighbors in X of x, and 0 if otherwise. The neighborhood vector $n(x)$ and x can be concatenated into an augmented vector $a(x) = \begin{bmatrix} x^T & n(x)^T \end{bmatrix}^T$. Using the augmented vector, we can easily combine the structural similarity and pairwise similarity, because $k_{\mathrm{STR}}(x_i, x_j) = n(x_i)^T n(x_j)$ and consequently

$$k_{\mathrm{LIN}}(a(x_i), a(x_j)) = k_{\mathrm{STR}}(x_i, x_j) + k_{\mathrm{LIN}}(x_i, x_j) . \tag{3}$$

It is also possible to trade-off the importance between the two similarity terms by defining $a(x)$ as $a(x) = \begin{bmatrix} x^T & \lambda n(x)^T \end{bmatrix}^T$ and varying the value of λ.

[1] We will assume that $x_i \mapsto x_i$ is always true.

Algorithm 1. SVM learning algorithm utilizing the structural similarity

1: **Input**: k, λ, and a training set $(x_i, y_i), i = 1, \ldots, n, y_i \in \{-1, +1\}$.
2: **Training**:
3: Find the k-NN of every example x_i and augment x_i to $a(x_i) = [x_i^T \ \ \lambda n(x_i)^T]^T$;
4: Train a linear or histogram intersection SVM using augmented examples $(a(x_i), y_i)$.

5: **Testing**:
6: For any testing example q, find its k-NN and augment it as $a(q) = [q^T \ \ \lambda n(q)^T]^T$;
7: Applying the trained SVM model to $a(q)$ to get classification result for q.

Note that $n(x)$ equals to one row of G if $x \in X$. However, $n(x)$ is well-defined even if $x \notin X$. Thus k_{STR} can be easily applied in inductive learning methods. For example, in the SVM learning we first construct $a(x)$ for every training example x. We can then train an SVM using $a(x)$ instead of x. During the testing phase, any testing example q is converted to $a(q)$ and the trained SVM model can be readily applied. The SVM training and testing method that utilizes structural similarity is summarized in Algorithm 1.

It is worth noting that any kernel or distance measure can be used to find the nearest neighbors and to generate the augmented vectors $a(x)$ (line 3 and line 6 of Algorithm 1). It is not necessary to use the same kernel to generate the augmented vectors and to train the SVM (line 4 of Algorithm 1).

Since $\min(x_1, x_2) = x_1 x_2$ if $x_1, x_2 \in \{0, 1\}$, we have $k_{\mathrm{HI}}(a(x_i), a(x_j)) = k_{\mathrm{STR}}(x_i, x_j) + k_{\mathrm{HI}}(x_i, x_j)$. Thus the histogram intersection kernel $k_{\mathrm{HI}}(x_1, x_2) = \sum_j \min(x_{1j}, x_{2j})$ is able to combine structural similarity with pairwise similarity. The RBF kernel, however, does not have this property.

3 From Global to Local Balance

The structural similarity kernel k_{STR} can play several roles in SVM classification. Besides adding structural similarity in addition to pairwise similarity (and introducing non-linearity into linear SVM classification), it also introduces a local balance heuristic to the dual variables.

Given a set of labeled training examples $(x_i, y_i), i = 1, \ldots, n$, a soft margin linear SVM with the optimal separating hyperplane $w^T x + b = 0$ solves the following optimization problem [3]:

$$\begin{aligned} \min_{w,b} \ & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \ & y_i(w^T x_i + b) \geq 1 - \xi_i, \ \ \xi_i \geq 0, \forall\, i\ . \end{aligned} \tag{4}$$

The primal form Eq.4 has a corresponding dual form

$$\min_{\alpha} \ f(\alpha) = \frac{1}{2}(\alpha \odot y)^T K(\alpha \odot y) - e^T \alpha \tag{5}$$

$$\text{s.t.} \ \sum_{i=1}^{n} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, \forall\, i\ , \tag{6}$$

where $y = [y_1, \ldots, y_n]^T$, $e = [1, \ldots, 1]^T$, $K_{ij} = x_i^T x_j$, and $\odot$ is the element-wise product operator. We will refer to Eq. 6 as the *global balance constraint*, as it specifies that the sum of dual variables in the two classes must be equal to each other in the entire training set. As observed in [16], in an imbalanced problem (where the majority/negative class has much more examples than the minority/positive class), Eq. 6 usually leads to more support vectors in the majority class than in the minority one. It is stated in [16] that a positive example close to the boundary are more likely to be classified as negative due to the global balance constraint Eq. 6.

When we train on the augmented examples $a(x_i)$ instead of x_i, the dual objective function now becomes (with the constraints remain unchanged)

$$f'(\alpha) = f(\alpha) + \frac{1}{2}\|s\|^2 \,, \tag{7}$$

where we define $s = G^T(\alpha \odot y)$. In addition to minimizing the usual SVM objective function $f(\alpha)$, the term $\|s\|^2$ imposes another heuristic that encourages s_j to be 0 (or small) for all $j = 1, \ldots, n$.

Since

$$s_j = \sum_{i=1}^{n} G_{ij}\alpha_i y_i = \sum_{i:x_i \mapsto x_j} \alpha_i y_i \,, \tag{8}$$

it measures the balance in the local neighborhood $\{x_i : x_i \mapsto x_j\}$ related to the example x_j. In other words, $s_j = 0$ means that the dual variables in this local neighborhood is balanced between the positive and the negative class, which we term as the *local balance*. Local balance is a desired property when the training set is imbalanced. If the local balance property is strictly satisfied (i.e., $s_j = 0$ for all j), it is guaranteed that for any support vector in the negative class, there will be at least one support vector in the positive class in a local neighborhood. The weights (dual variables α_i) of these support vectors are balanced. Thus, even a positive test example near the boundary will not be dominated by negative examples.

The global balance property Eq. 6 can be safely neglected if local balance is strictly satisfied for all support vectors. These two balance properties are linked by the following relationship:

$$\sum_{j=1}^{n} s_j = \sum_{j=1}^{n} \sum_{i:x_i \mapsto x_j} \alpha_i y_i = \sum_{i=1}^{n} \alpha_i y_i \left(\sum_{j:x_i \mapsto x_j} 1 \right) = k \sum_{i=1}^{n} \alpha_i y_i \,. \tag{9}$$

The last equality in Eq. 9 uses the fact that Γ_X is a kNN graph and the out-degree of any node x_i is k. When $s_j = 0$ for all j (which implies a less strict condition $\sum_j s_j = 0$), global balance is automatically satisfied.

From now on we will only consider support vector machines that do not have the bias term b (and consequently without the global balance constrain Eq. 6). If a bias term is needed, one can append to each example an additional feature dimension with constant value 1.

4 From Hub Scores to Signed Authority Scores

The form of the definition $s \equiv G^T(\alpha \odot y)$ reminds us of the equation that turns hub scores into authority scores in the HITS algorithm [9]. This resemblance leads to an alternative view of Algorithm 1.

The HITS algorithm finds authoritative sources in a hyperlinked environment (i.e., directed graph). For example, for a text-based web search "car maker", authorities are those nodes (webpages) that contain useful information about car manufacturers. Authoritative nodes are expected to have a large number of incoming edges. Hubs are defined in [9] as those nodes that have many outgoing edges. The authority score v and hub score u of all nodes are then iteratively updated by the relationship $v = G^T u$ and $u = Gv$, where G is the adjacency matrix of the directed graph.

The HITS algorithm is an unsupervised method. In our supervised learning settings, the vector $\alpha \odot y$ acts as signed hub scores. The optimized dual variables α contain the weights or relative importance levels of the training examples. This vector is analogous to the hub score vector u in the HITS algorithm. The vector $\alpha \odot y$ is a natural extension of the hub scores into supervised learning. The local balance vector $s = G^T(\alpha \odot y)$ is then a good candidate for the supervised extension of the authority score vector v.

Let us consider a simplified problem where we only take into account the structural similarity. An example x_i is converted into the neighborhood vector $n(x_i) = G_{i:}$, where $G_{i:}$ is the i-th row of the adjacency matrix G. A linear SVM using $n(x_i)$ as training examples will lead to a classification boundary that is

$$w = \sum_{i=1}^{n} \alpha_i y_i n(x_i) = G^T(\alpha \odot y) = s \; . \tag{10}$$

In other words, the local balance vector s equals the classification boundary w if we only consider the structural similarity. In this scenario, the local balance vector s contains in effect the signed authority scores learned through the SVM optimization. When a test example q is given, the neighborhood vector $n(q)$ is a sparse vector with the value 1 in j-th position only if x_j is within the k nearest neighbors of q. Thus the decision value for q is $n(q)^T w = n(q)^T s = \sum_{j:q \mapsto j} s_j$. We just need to find the k nearest neighbors of q, and they will each contribute a local signed authority score s_j. The classification of q is then determined by the sum of these k local signed authority scores. This fact exhibits an important property of structural similarity in SVM learning: during the testing time, only the nearest neighbors of the query q will affect the classification result. Support vectors that are far away from q will not affect the decision on q. This property is advantageous in learning imbalanced problems.

When we use the complete augmented vectors $a(x_i) = [x_i^T \;\; n(x_i)^T]^T$ to train a linear SVM, the last n dimensions of the resulting classification boundary will correspond to s. The decision value for q is then $\sum_{j:q \mapsto j} s_j + \sum_{i=1}^{n} \alpha_i y_i x_i^T q$. The second term can also be expressed in terms of local signed authority scores. Denoting $q_X = [x_1^T q, \ldots, x_n^T q]^T$, the second term can be written as $(\alpha \odot y)^T q_X = s^T(G^{-1} q_X)$.

5 Related Research

Local geometry has long been utilized in machine learning, e.g., in (unsupervised) dimensionality reduction (LLE [12]). A neighborhood graph (in which nodes and edges denote examples and similar example pairs respectively) is a common way to utilize the local geometry information (e.g., PageRank [10]). In Kleinberg's HITS algorithm [9], link analysis techniques are applied on the neighborhood graph to find authoritative and hub webpages in an iterative and unsupervised manner (c.f. Sec. 4). In webpage classification, natural link structures exist and bibliographic coupling has been used to extract features [11].

Linking structures are particularly useful when the pairwise similarity computed only using the feature vectors of two examples are not accurate. For example, in computer vision and image analysis, visual features frequently suffer from the fact that their intra-class variation is bigger than the inter-class variation. Graph structural similarity have been successfully used to unsupervisedly recognize objects [7] and localizing regions of interest [8]. The structural similarity in [7] was computed using a method that combines both bibliographic coupling and co-citation (refer to [1]), and is computationally more expensive than Eq. 2. The co-citation count between nodes x_i and x_j is the number of nodes that have edges pointing to both x_i and x_j, i.e., reversing the edge directions of the bibliographic couping count in Eq. 2.

Graphs that encode local geometry also form a major thread of methods for semi-supervised learning [19]. In semi-supervised learning, the nodes are examples (may be unlabeled) and edges connect similar pairs of nodes. A graph is usually coupled with a function f which must be close to the labels of labeled nodes, and is smooth on the entire graph [19]. Thus the graph is acting as a smoothing constraint for the problem.

In particular, a global smoothness functional based on normalized co-citation counts was used in [18]. Another closely related semi-supervised method [13] defines a family of data-dependent norms, which can warp the structure of a reproducing kernel Hilbert space to the geometry of both labeled and unlabeled data. Thus a supervised kernel learning method can be turned to learn from both labeled and unlabeled data, and can be applied to test unseen examples. The smoothness assumption in [13] is implemented through the graph Laplacian, and requires inverting of a $n \times n$ matrix where n is the total number of examples.

6 Experiments

6.1 Setup

Experiments on 6 datasets are used to test the proposed method. These datasets are `ijcnn1`, `protein`, `satimage`, `shuttle`, `splice`, and `vowel`. We used the scaled version of these datasets downloaded from the LIBSVM dataset page.[2]

[2] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Baseline SVM accuracies are provided using the RBF, linear, and histogram intersection kernel (columns `rbf`, `lin`, and `hik` in Table 1, respectively). SVM-weight with RBF kernel is used as a baseline imbalanced learning SVM method (column `rbf-w`), which (although simple) has been shown to be very effective in imbalanced learning [15]. In SVM-weight, different costs are associated with errors in the minority class (C_+) and the majority class (C_-). We set $\frac{C_+}{C_-} = \frac{n_-}{n_+}$, where n_+ and n_- are the number of examples in the minority and majority classes, respectively.

As the structural similarity kernel can be seamlessly combined with the linear and histogram intersection kernel (Algorithm 1), the proposed method is presented for these two kernels. Since the kernel (or distance measure) used to generate a kNN graph needs not to be the same as the kernel used in SVM training, we use a notation "X+Y": 'X' represents the kernel for generating the neighborhood kNN graph and augmented feature vectors, and 'Y' represents the kernel for subsequent SVM training. Note that the γ parameter in the RBF kernel will not affect the kNN graph when 'X' equals RBF. In this case, we simply ignore the exponential function and the γ parameter when building a kNN graph for better numerical stability.

The LIBSVM package [2] is used for RBF kernel SVM (which uses the 1-vs-1 strategy for multi-class classification). The LIBLINEAR package [5] is used for linear SVM methods. Although the histogram intersection kernel (HIK) is not a positive definite kernel for real-valued feature vectors. It is proved to be a positive definite kernel when feature values are non-negative numbers, and achieves better performances than the linear kernel in most cases [17]. For HIK SVM, we linearly quantize all feature values into the range $[0 \;\; 100]$, and use the fast HIK SVM method proposed in [17]. The 1-vs-all strategy is used in both LIBLINEAR and [17]. SVM parameters (C in all methods, and γ in RBF SVM) are searched using 5-fold cross-validation on the training set, in the range $\log_2 C \in [-11 \;\; 15]$ and $\log_2 \gamma \in [-11 \;\; 3]$, with the search grid size being 2.

In an augmented feature vector $a(x) = [x^T \;\; \lambda n(x)^T]^T$, the parameter λ controls the trade-off between the pairwise similarity and the structural similarity. In linear SVM, we simply set $\lambda = 1$. In HIK SVM, the features x are quantized to $[0 \;\; 100]$, and we set $\lambda = 10$. The number of nearest neighbors k is set to 10 for all experiments.

6.2 Results

Three accuracy numbers are reported for every method. In every cell of Table 1 the first number is the overall accuracy (number of correctly predicted examples divided by total number of testing examples, `acc`) in percentage. The second number is the arithmetic average (`a-mean`) accuracy of all classes (i.e., average of the diagonal entries of the confusion matrix). The third number is the geometric average (`g-mean`) accuracy of all classes, which is a commonly used performance measure in imbalanced learning.

Note that `g-mean` is always less than or equal to `a-mean`. And we usually observe that `g-mean` $<$ `a-mean` $\ll$ `acc` in imbalanced learning problems. Although

Table 1. Comparing accuracies of baseline SVM and SVM that integrates structural similarity, in which `a-m` and `g-m` mean `a-mean` and `g-mean`, respectively. `rbf-w` means SVM-weight with the RBF kernel.

	rbf			lin			lin+lin			rbf+lin		
	acc	a-m	g-m	acc	a-m	g-m	acc	a-m	g-m	acc	a-m	g-m
ijcnn1	98.03	92.22	91.94	91.71	63.15	52.38	96.13	86.31	85.45	97.09	90.61	90.25
protein	69.19	65.79	65.07	68.98	64.34	62.96	68.07	63.55	62.22	69.10	64.83	63.68
satimage	89.30	86.98	85.64	81.60	74.73	57.11	86.80	83.10	80.21	89.70	87.67	86.62
shuttle	99.68	69.02	00.00	91.82	41.31	00.00	98.86	67.17	57.72	99.88	92.38	91.73
splice	89.66	89.70	89.70	84.97	84.88	84.86	83.86	83.82	83.81	86.76	86.79	86.78
vowel	54.98	54.98	51.90	41.56	41.56	36.58	45.02	45.02	41.30	50.43	50.43	45.79
Avg.	83.47	76.45	64.04	76.77	61.66	48.98	79.79	71.49	68.45	82.16	78.79	77.48
	rbf-w			hik			hik+hik			rbf+hik		
ijcnn1	96.56	93.86	93.80	94.88	84.43	83.44	97.08	92.37	92.18	97.60	93.47	93.33
protein	68.90	65.83	65.22	68.81	65.40	64.69	69.94	66.52	65.82	69.20	65.83	65.14
satimage	87.70	87.40	87.02	89.10	87.10	85.93	89.35	87.44	86.41	89.70	87.79	86.78
shuttle	97.94	96.24	96.08	99.01	85.46	83.74	99.92	93.77	93.04	99.90	95.23	94.83
splice	89.66	89.70	89.70	93.43	93.51	93.49	93.89	93.95	93.94	93.84	93.93	93.90
vowel	54.98	54.98	51.90	41.13	41.13	35.60	50.22	50.22	45.63	55.19	55.19	52.68
Avg.	82.62	81.34	80.62	81.06	76.17	74.48	83.40	80.71	79.50	**84.24**	**81.91**	**81.11**

there are more sophisticated statistics to measure imbalanced learning performance (e.g., area under the Prevision-Recall or ROC curve [4]), the two simple average accuracy measures are sufficient to clearly show the differences in Table 1. Also note that both the arithmetic and geometric mean apply to problems with multiple classes.

As shown in Table 1, *incorporating the structural similarity improves SVM classification accuracies in most cases*, no matter whether a linear or histogram intersection kernel is used. In the linear SVM, the neighborhood vectors $n(x)$ extracts non-linear features from x and increases the capacity of the classifier. The mean accuracies on the 6 datasets increases from 76.77% (linear SVM) to 82.16% by adding structural similarity (RBF kNN graph plus linear SVM). The `rbf+lin` method achieves higher accuracies than a simple linear SVM in all datasets and all three measures. Using the linear kernel to generate kNN graph and structural similarity is not as effective as the RBF kernel. However, `lin+lin` still achieves higher accuracies than `lin` in 4 out of 6 datasets. It is worth noting that `lin+lin` usually outperforms `lin` by a large margin (4-7%). But when `lin` have higher accuracies than `lin+lin`, the differences are usually only about 1%.

Similarly, structural similarity also boosts performance of the HIK SVM. Both `rbf+hik` and `hik+hik` outperforms `hik` in all datasets and all performance measures. The absolute level of improvements by structural similarity in HIK SVM, however, is smaller than that in linear SVM. One reason might be that the HIK SVM itself has higher discriminative power than the linear SVM. It is worth noting that `rbf+hik` has better `acc` performances than RBF SVM in almost all datasets, except in the `ijcnn1` dataset where `rbf+hik` is slightly worse.

In fact, when structural similarity (RBF kNN graph) is used, even the linear SVM performs closely to the RBF kernel SVM.

Structural similarity, however, is *most effective in dealing imbalanced problems*. In plain SVM methods, the `a-mean` measure is usually much higher than the `g-mean` measure. After equipped with the structural similarity, the SVM classifiers have similar `a-mean` and `g-mean` values. For example, in `rbf+hik`, the difference is smaller than 1%. This phenomenon means that the accuracy in different classes are close to each other, based on the inequality of arithmetic and geometric means, i.e., *the proposed imbalanced learning strategy is indeed effective*. The average `a-mean` measure is improved from 61.66% (linear SVM) to 78.79% (`rbf+lin`), and from 76.17% (HIK SVM) to 81.91% (`rbf+hik`) in these datasets. We can observe even larger improvements of the `g-mean` measure in Table 1.

The `shuttle` dataset is the most imbalanced one, with only 2 training examples in the smallest class and 11478 examples in the largest class. The RBF kernel SVM has an `a-mean` accuracy of 69.02%, in which the 4 classes with less than 50 training examples are almost completely misclassified (in consequence the `g-mean` measure is 0.). The linear and histogram intersection kernel are not effective in dealing with this highly imbalanced problem either. Structural similarity, however, successfully recognizes examples from these minority classes. For example, `rbf+hik` has an `a-mean` of 95.23% and a `g-mean` of 94.83%. Specifically, `lin` has 0% accuracies on all these 4 minority classes. However, the accuracy of `rbf+lin` for these 4 classes are 76.92%, 94.87%, 75%, and 100% , respectively.

In Table 1 we also compare the proposed method with SVM-weight, a simple but effective imbalanced SVM learning method. The `rbf+hik` method outperforms `rbf-w` in all three performance measures. Although SVM-weight is also effective in balancing the accuracies between the minority and majority classes, it has lower average `a-mean` and `g-mean` values than `rbf+hik`. We want to emphasize that imbalanced learning methods such as SVM-weight usually increase measures such as `g-mean` at the cost of a reduced accuracy `acc`. However, *the proposed method carries out effective imbalanced learning without hurting the classification accuracy* `acc`. In fact, `rbf+hik` has a higher average `acc` value than that of SVM classifiers with linear, histogram, and RBF kernels, and the SVM-weight classifier.

We end our discussions about Table 1 with a note about the `splice` dataset. This dataset is almost balanced (the number of examples in the minority class is close to that of the majority class). `rbf-w` and `rbf` achieve exactly the same accuracies on this dataset. However, the similarity kernels can still improve all the three performance measures over the plain SVM in `rbf+lin`, `hik+hik`, and `rbf+hik`.

6.3 Discussions of Limitations

In spite of its effectiveness, the structural similarity kernel requires a time-consuming step to construct a kNN graph. At the testing phase, finding the k-nearest neighbor of a query is also computationally expensive. On the other

hand, because fast SVM training algorithms are available for both the linear and the histogram intersection kernel SVM, the time for cross-validation parameter selection and SVM training can be significantly reduced. For example, the total time for parameter selection, training, and testing is 4524 seconds for RBF kernel SVM on the `shuttle` dataset. The total time for `rbf+lin` and `rbf+hik` are only 769 seconds and 1028 seconds, respectively.

Another issue is the choice of λ. A too small or too large λ will effectively reduce the augmented vector $a(x)$ to either the original feature vector x or the neighborhood vector $n(x)$. Although the experiments in Table 1 indicates that the default choice of λ leads to reasonable results, it is worthwhile to learn a suitable λ for each dataset. For example, x and $n(x)$ can be used to construct two kernel matrices separately. By formulating structural similarity SVM as a multiple kernel learning problem, MKL methods can be used to learn the value of λ. An additional advantage is that kernels besides the linear kernel and the histogram intersection kernel can be used together with the structural similarity kernel (e.g., `X+rbf`).

7 Concluding Remarks

A structural similarity kernel is presented in this paper for SVM learning, and in particular when the problem is imbalanced. Kernels like the RBF kernel computes the similarity of two examples using only the feature vectors of them. However, after building a neighborhood graph (kNN graph), the linking patterns of two examples (nodes) convey useful information about how similar they are. The proposed structural similarity kernel captures this structural similarity by computing the bibliographic coupling count, and its feature space representation corresponds to rows of the adjacency matrix of the kNN graph. The structural similarity kernel is a data-dependent kernel.

The structural similarity kernel can be seamlessly integrated into the linear SVM or the histogram intersection kernel SVM. We show that it is equivalent to adding a regularization term that encourages balanced weights in all local neighborhoods defined by incoming edges in the kNN graph. Analogous to the unsupervised HITS algorithm, the structural similarity kernel turns hub scores into signed authority scores, and is particularly effective in dealing with imbalanced learning problems. Experimental results on several datasets show that structural similarity can help linear and histogram intersection kernel to match or surpass the performance of the RBF kernel in terms of classification accuracy. When the problem is imbalanced, structural similarity can significantly improve imbalanced learning performance measures such as `g-mean`, while at the same time it still maintains high classification accuracy.

Further research that will improve the structural similarity kernel SVM including at least two directions: a fast (maybe approximate) way to create the kNN graph, and the use of multiple kernel learning to learn an appropriate λ parameter value.

Acknowledgement. J. Wu is supported by the NTU SUG grant and AcRF Tier 1 grant RG 34/09.

References

1. Blondel, V.D., Gajardo, A., Heymans, M., Senellart, P., Dooren, P.V.: A measure of similarity between graph vertices: Applications to synonym extraction and web searching. SIAM Review 46(4), 647–666 (2004)
2. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), http://www.csie.ntu.edu.tw/~cjlin/libsvm
3. Cortes, C., Vapnik, V.N.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
4. Davis, J., Goadrich, M.: The relationship between Precision-Recall and ROC curves. In: Int'l Conf. on Machine Learning, pp. 233–240 (2006)
5. Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S.S., Sundararajan, S.: A dual coordinate descent method for large-scale linear SVM. In: Int'l Conf. on Machine Learning, pp. 408–415 (2008)
6. Kessler, M.M.: Bibliographic coupling between scientific papers. American Documentation 14(1), 10–25 (1963)
7. Kim, G., Faloutsos, C., Hebert, M.: Unsupervised modeling of object categories using link analysis techniques. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (2008)
8. Kim, G., Torralba, A.: Unsupervised detection of regions of interest using iterative link analysis. In: Advances in Neural Information Processing Systems (2009)
9. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
10. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1998)
11. Qi, X., Davison, B.D.: Web page classification: Features and algorithms. ACM Computing Surveys 41(2) (2009)
12. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
13. Sindhwani, V., Niyogi, P., Belkin, M.: Beyond the point cloud: from transductive to semi-supervised learning. In: Int'l Conf. on Machine Learning, pp. 824–831 (2005)
14. Swain, M.J., Ballard, D.H.: Color indexing. International Journal of Computer Vision 7(1), 11–32 (1991)
15. Tang, Y., Zhang, Y.-Q., Chawla, N., Krasser, S.: SVMs modeling for highly imbalanced classification. IEEE. Trans. Systems, Man, and Cybernetics, Part B: Cybernetics 39(1), 281–288 (2009)
16. Wu, G., Chang, E.Y.: Class-boundary alignment for imbalanced dataset learning. In: ICML 2003 Workshop on Learning from Imbalanced Data Sets (2003)
17. Wu, J.: A fast dual method for HIK SVM learning. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 552–565. Springer, Heidelberg (2010)
18. Zhou, D., Schölkopf, B., Hofmann, T.: Semi-supervised learning on directed graphs. In: Advances in Neural Information Processing Systems (2004)
19. Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin, Madison (2005)

Using Classifier-Based Nominal Imputation to Improve Machine Learning

Xiaoyuan Su[1], Russell Greiner[2], Taghi M. Khoshgoftaar[1], and Amri Napolitano[1]

[1] Computer and Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431, USA
{suxiaoyuan,amrifau}@gmail.com, taghi@cse.fau.edu
[2] Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8
rgreiner@ualberta.ca

Abstract. Many learning algorithms perform poorly when the training data are incomplete. One standard approach involves first imputing the missing values, then giving the completed data to the learning algorithm. However, this is especially problematic when the features are nominal. This work presents "classifier-based nominal imputation" (CNI), an easy-to-implement and effective nominal imputation technique that views nominal imputation as classification: it learns a classifier for each feature (that maps the other features of an instance to the predicted value of that feature), then uses that classifier to predict the missing values of that feature. Our empirical results show that learners that preprocess their incomplete training data using CNI using support vector machine or decision tree learners have significantly higher predictive accuracy than learners that (1) do not use preprocessing, (2) use baseline imputation techniques, or (3) use this CNI preprocessor with other classification algorithms. This improvement is especially apparent when the base learner is instance-based. CNI is also found helpful for other base learners, such as naïve Bayes and decision tree, on incomplete nominal data.

Keywords: incomplete data, imputation, support vector machine, instance-based learning, nominal data.

1 Introduction

It is often difficult to learn good classifiers when the training data are missing attribute values. To deal with missing data in classification tasks, many learners first use some imputation technique to fill in the missing values, before giving the completed data to a complete-data learner. A simple imputation technique is to replace each missing value of a real-valued attribute with the mean of the observed values of the attribute (MEI), or a nominal attribute with its most commonly observed value (MCI). This is used by the WEKA implementations for many classification algorithms [4]. However, these trivial imputers generally do not help produce high-quality classifiers for incomplete data.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 124–135, 2011.

Many learning algorithms have algorithm-specific built-in schemes for handling missing value – e.g., naïve Bayes can simply ignore the missing attributes at both learning and classification time, and some instance-based algorithms simply set the distance measure to any attribute (of an instance) missing an entry to the associated maximum value [1]. However, these simple missing data handling methods often do not produce accurate estimates to fill in the missing values. As we anticipate that answers based on accurately imputed data will be better than those based on the original incomplete data and on less accurately imputed data, we expect preprocessing the incomplete data with accurate imputers can boost the classification performance of machine learners on incomplete data. In general, an "imputation-helped learner" uses some imputation techniques to fill in the missing values before training a classifier. Su *et al.* investigated such imputation-helped learners in the context of numeric features, and showed that certain numerical imputation techniques can improve classification performance [9]. However, few existing works specifically investigate how imputation techniques can improve classification performance on nominal data.

This work explores imputation-helped learners for nominal features. We first propose an easy-to-implement algorithm for imputing nominal features, *classifier-based nominal imputation (CNI),* which treats imputation as a classification task: first learn a classifier for each feature, then use this trained classifier to impute values for the missing entries. We let the notation "kNN-CNI(SVM)" refer to the learning system that preprocesses the incomplete training data by learning a SVM classifier for each attribute to impute each missing value of this attribute (so if there are n attributes with missing values, this produces n different classifiers); the resulting completed dataset from this CNI(SVM) preprocessing is then given to the learner kNN ("k nearest neighbors" [1]) to produce a final classifier. In general, the "B-CNI(L)" learner first uses the learner L to learn n different classifiers for imputing values for the n attributes, then gives the completed data to the base learner B.

We first investigate the imputation performance of our proposed CNI imputation using 10 machine learners as imputation classifiers, and found that (1) each of these CNI imputers has more accurate predictions than the baseline kNN imputation and MCI, and (2) CNI(SVM) (classifier-based nominal imputation that uses support vector machine) and CNI(DT) (that uses decision tree) perform especially well.

By applying the top-performing CNI imputers to preprocess incomplete nominal data, our empirical experiments show that these imputation techniques can significantly improve classification performance for instance-based algorithms on incomplete nominal data with either a high or low percentage of missing values. While imputation is not as critical for other learning algorithms, such as naïve Bayes and decision tree, we found that our proposed B-CNI(L) approach can still boost their classification performance when the missing ratio is at or below 20%.

Section 2 describes the framework of this paper, Section 3 provides our experimental design and results, and Section 4 contains the conclusions.

2 Framework

2.1 Imputation for Nominal Data

As many real-world datasets are missing some information, imputation techniques are often used to fill in the missing values with estimated values; this often leads to performance that is better than just using the original incomplete data. Imputation techniques for numeric data, such as EM (expectation maximization) and BMI (Bayesian multiple imputation), involve iteratively updating estimates of means and covariance matrices, as the data is assumed to be normally distributed [9]. This is, of course, typically not appropriate for nominal data.

A baseline imputation for such nominal data is MCI (most common [value] imputation), which fills the missing values with the most frequently observed value of the attribute. However, MCI distorts the distribution of the values by overestimating the most frequent value, which often leads to incorrect inferences. Figure 1 shows that MCI does not produce a similar shape of attribute value distribution as the original missing data, while our proposed CNI imputation does; see details in the next subsection.

Another well-known nominal imputation technique is kNN imputation (kNNI) [2], which imputes a missing value of an attribute in an instance as the most common value of that attribute in the instance's k nearest neighbors. However, kNNI is not very effective because of the way that kNN selects the nearest neighbors: this is based on a distance function that is problematic in the presence of incomplete data (see Section 2.3 below).

We therefore propose an easy-to-implement nominal data imputation technique: classifier-based nominal imputation. This paper investigates whether this nominal imputation technique can be used to improve classification performance for machine learned classifiers on incomplete nominal data.

2.2 Classifier-Based Nominal Imputation

The basic idea of classifier-based nominal imputation (CNI) is simple: treat imputation as classification. For each attribute f_i with missing values, learn a classifier $c_i(\ldots)$ that takes as input the values of the other $n-1$ attributes $\{f_j | j \neq i\}$ for an instance, and returns the value for f_i for this instance. CNI(L)

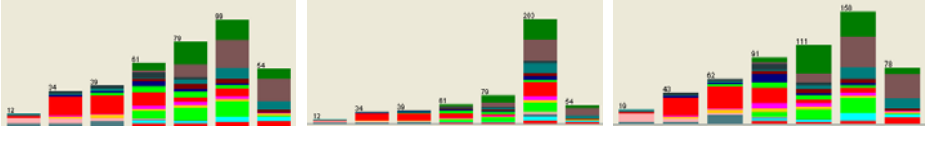


Fig. 1. Attribute value distributions on the "date" attribute of the Soybean-large dataset with 30% of the values missing: (from left to right) (1) the original missing data, (2) after most common imputation (MCI), (3) after CNI using decision tree imputation CNI(DT)

uses the learning algorithm L to learn this $c_i(\ldots)$ classifier from the training instances with observed values on f_i; CNI then uses this $c_i(\ldots)$ to impute the missing values of this attribute in the remaining instances. Algorithm 1 illustrates the CNI imputation algorithm.

After imputing incomplete data using CNI(L), B-CNI(L) passes the resulting completed dataset to the base learner B, which produces a classifier that can then classify (possibly incomplete) test instances. Note that the L learner must be able to deal with incomplete data as its data sample D_i^+ (see Algorithm 1) will typically have missing values, and the $c_i(\ldots)$ classifier that it produces must also be able to handle missing information. However the base learner B will only need to deal with complete instances. The resulting classifier will predict class labels for the original incomplete test data.

Note that the values imputed for one attribute could be used in the later iterations of imputation. For example, if attribute f_1 was the first imputed attribute, its imputed values could be used when imputing values for f_2 and for all other subsequent attributes. However, to simplify the algorithm and focus on the general techniques, we did not use the previously imputed values for any of the later attributes in this work.

Here, we investigate the following Imputation Learners L: decision tree (C4.5), decision table (dTable), lazy Bayesian rules (LBR), naive Bayes (NB), one rule (OneR), decision list (DecList), random forest (RF), support vector machine (SVM), radial basis function neural networks (RBF), and multilayer perceptron neural network (MLP). Each of the learners and the CNI technique are implemented within the WEKA [4] framework, and use WEKA's default approach for dealing with missing values, at both learning and performance time. When there are too few observed feature values to train reasonable classifiers (here, under three instances with observed values for attribute f_i in D_i^+), we simply use MCI to impute a value for that attribute. For comparison, we also implement the baseline nominal imputation techniques kNNI and MCI.

Algorithm 1. CNI(ImputationLearner L)

for $t = 1 \ldots n$ **do** % over each feature (attribute column of dataset D)
  % View *feature* f_i as the class and the other columns as features
  Divide instances $D = D_i^+ + D_i^-$
    where training set D_i^+ contains the instances with observed values of attribute f_i, and test set D_i^- contains the instances that miss values of attribute f_i
  Let $c_i(\ldots) = L(D_i^+)$ be classifier trained using learner L on D_i^+
    using L's default approach to handle missing data, as necessary
  for each instance $d \in D_i^-$ **do**
    Let d_{-i} be the $n-1$ "non-f_i" values in d
    Impute the "f_i" value of d as $c_i(d_{-i})$
  % Move on to the next attribute
% return the resulting completed dataset D'

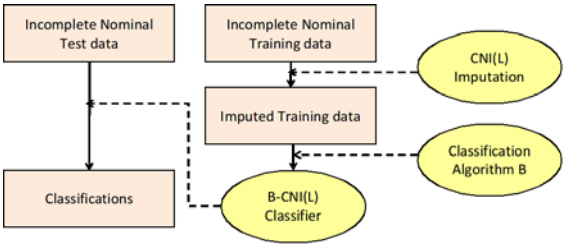


Fig. 2. The B-CNI(L) framework of improving the classification performance of machine learning algorithm B using L imputation as the preprocessor for the nominal training data

As these classifiers are well-known, we do not provide detailed descriptions for each of them here. We use default parameter settings of WEKA for L and B learners unless they are explicitly specified otherwise. For the imputation learner L=SVM, we use the linear kernel [7].

2.3 Using CNI to Improve Classification Performance of Machine Learned Classifiers

Figure 2 shows the general B-CNI(L) framework, which first uses some nominal learner L for imputing the missing nominal values in the training set to generate an imputed training dataset, then gives the completed dataset to a learning algorithm B (e.g., kNN) to learn a classifier. We then use this classifier to classify the (possibly incomplete) test instances.

Although it is legitimate to impute training data together with test data (excluding the labels of the test data), our imputers only work on the training data, and use the original incomplete test data for evaluation. (When predicting class labels for each incomplete test instance, we use the missing data handling strategy of the classifier associated with the base learner B). We use this imputation scenario for dealing with incomplete training/test data because, in practice, training data and test data often come in different times. Therefore, it would be impractical to impute training data and test data together in many cases.

We investigate how our proposed CNI imputation can help machine learners such as instance-based learning algorithms, naïve Bayes, decision tree, and neural network.

Instance-based learning is a kind of "lazy learning" that assigns a label to a new unlabeled instance based on the "closest" labeled instances appearing in the training set. A commonly used instance-based learning algorithm is the k-nearest neighbor algorithm, which first identifies the k neighbors nearest to the "to be labeled instance" (based on the distance values calculated between this new instance with each of the instances in the training set) and returns the majority class over these neighbors, or perhaps some variant that weights the labels of these neighbors. The distance function in instance-based learning is usually defined as [1]:

$$Dis(x,y) = \sqrt{\sum_{i=1}^{n} f(x,y)} \quad (1)$$

where $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ are instances, each over n attributes. For numeric attributes, $f(x_i, y_i) = (x_i - y_i)^2$, and for Boolean and nominal attributes, $f(x_i, y_i) = I(x_i \neq y_i)$ which is 0 if $x_i = y_i$ and is 1 otherwise. The nearest neighbor for an instance x is $argmin_y Dis(x, y)$ over all instances y in the training set.

Of course, Equation 1 is not defined if any $f(x_i, y_i)$ value is undefined. However, as $f(\ldots)$ must return an answer even if either x_i or y_i is missing, many nearest-neighbor systems will use extreme values – ie, set $f(x_i, y_i)$ to the maximum value if either x_i or y_i is missing [1]. As this simple approach often leads to biased distance values, we suspect that a reasonable estimate of missing values in the training set will improve the resulting classifier.

The best value of k often depends on the dataset. In general, a larger k value reduces noise on the classification, but can impair performance if it is too big. A good k can be selected by using various techniques, such as cross-validation or some heuristic. We often consider $k > 1$ nearest neighbors and set k to an odd value to prevent ties for binary classification. (For multiple classes, we use the plurality of the votes.) Some instance-based classifiers will also weight each neighbor in this vote by using $1/Distance$ or $1 - Distance$, both of which give higher weights to nearer neighbors [4].

When using imputation to help instance-based algorithms, imputing the training set will replace the missing values for the instances in the training set, while the missing values that have penalized distance values in the test instances are still present. Imputation reduces the number of times where instance-based learning algorithms will use the maximized distance values.

Many other learning algorithms deal naturally with incomplete data, and therefore imputation may not be so critically helpful. For example, naïve Bayes makes classifications based on only observed values [6], while C4.5, a well-known decision tree classifier, will in effect simply disregard the missing values during training [8].

3 Experimental Design and Results

We explored 12 nominal datasets from the UCI machine learning repository (see dataset description in Table 1) [3]. Six of the 12 datasets have both nominal and numeric attributes; these have *italicized* names in Table 1 – e.g., the dataset "Australian" has 8 nominal attributes out of a total of 14 attributes. Most of the nominal data have more than two attribute values. Here, when corrupting the data by removing values, we only remove some of the nominal attributes, and leave all of the numeric attributes.

To investigate the performance on datasets with different missing ratios, we generate five incomplete datasets for each of the above datasets by randomly

deleting 10%, 20%, 30%, 40%, and 50% of the observed values – this is done by removing each [instance, attribute] independently, with probably 0.1 [resp., 0.2, . . . , 0.5].

We first evaluate our proposed CNI(L) imputation algorithms using different machine learned classifiers L; we then pick the top two CNI imputers in terms of their estimation accuracy (see next subsection) to impute the incomplete nominal training sets, before applying a base learner to train a classifier on the imputed training set, which we subsequently use to classify the incomplete instances in the test set. The classification results reported are the average of five cross validation folds.

We then evaluate how CNI imputation can improve the classification performance for machine learners kNN, naïve Bayes, decision tree, and MLP on incomplete nominal data.

3.1 Evaluation of CNI Imputation Algorithms

As we generated the incomplete datasets by removing values from the complete datasets, we have the ground truth for each missing value. Therefore, our nominal imputation algorithms can be evaluated in terms of the estimation accuracy by checking the imputed values against their respective ground truth values.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} I(P_i = R_i) \tag{2}$$

where P_i is the estimated value produced by the imputer (for some [instance, attribute] pair), R_i is the ground truth value, and N is the total number of the imputed values.

Figure 3 illustrates the imputation performance of the CNI imputers $L \in$ {SVM, C4.5, NB, RF}, kNNI and MCI, when 30% of the data is missing.

Table 1. Datasets from the UCI Machine Learning Repository

dataset	#instances	#attributes (nominal/all)	average attribute values	#classes
Chess	3196	35	2.03	2
Corral	128	6	2	2
Flare	1066	10	3.3	2
Mofn-3-7-10	1324	10	2	2
Soybean-large	562	35	2.86	19
Vote	435	16	3	2
Australian	690	8/14	4.63	2
Cleve	296	7/13	2.71	2
Crx	653	9/15	4.56	2
German	1000	13/20	4.31	2
Hepatitis	80	13/19	2	2
Lymphography	148	15/18	2.93	4

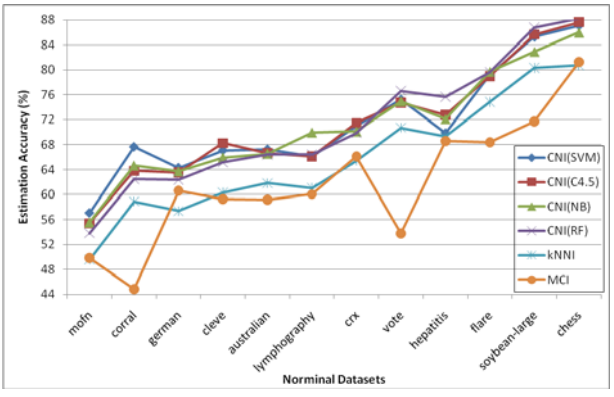


Fig. 3. Estimation accuracies of the CNI imputers, kNNI, and MCI on the datasets with a missing ratio of 30%

Our proposed nominal imputation algorithm – classifier-based nominal imputation (CNI) – performs significantly better than the commonly used nominal imputers, kNNI and MCI. While each CNI(L) imputer (that uses the learner L) we investigated outperforms kNNI and MCI, each of the top performers CNI(SVM) and CNI(C4.5) has more than 7% and 15% higher average estimation accuracies over kNNI and MCI respectively. Statistically, CNI(SVM) and CNI(C4.5) outperform kNNI with 1-sided t-test $p < 2 \times 10^{-8}$ and $p < 1 \times 10^{-8}$ respectively, and even the worst performing CNI(RBF) and CNI(MLP) still slightly outperform kNNI, albeit with $p < 0.01$ and $p < 0.1$. As expected, the estimation accuracies of the imputers decrease as the missing ratio of the datasets increases.

For all of the 10 nominal imputers we investigated, the ranking of the nominal imputers in terms of their average estimation accuracies over the 12×5 datasets (five datasets with different missing ratios between 10% and 50% generated from each of 12 base datasets) is: CNI(SVM), CNI(C4.5), CNI(DecList), CNI(NB), CNI(LBR), CNI(RF), CNI(OneR), CNI(dTable), CNI(RBF), CNI(MLP), kNNI, and MCI. We will therefore use the best two, CNI(SVM) and CNI(C4.5), as nominal imputers in the investigations below.

3.2 The Impact of Nominal Imputers on the Classification Performance for Instance-Based Learning Algorithms

Now we evaluate how much nominal imputers can improve the classification performance for instance-based learning algorithms.

We work on instance-based algorithms with different weighting schemes. Rather than setting the best k value for each different dataset, different nominal imputer and distance weighting scheme, we will instead simply use $k = 5$ for all cases.

Table 2 is a summary of the classification results, with cells that record the average classification accuracy over the 12 nominal datasets. Figure 4(a) illustrates

how nominal imputers improve the classification performance of instance-based algorithms on the dataset "Australian" over various different missing ratios.

Table 2 and Figure 4(a) show that instance-based algorithms decrease their classification accuracy fast with the increase of the missing ratio of data. With the help of nominal imputers, kNN can achieve a statistically significant improvement of classification accuracy (here, this is based on a 1-sided paired t-test, with $p < 0.05$; note all statistical claims are based on this test). This is true for all the datasets with all missing ratios we investigated (in the range of 10% and 50%), and for different distance weighting schemes of instance-based algorithms, including (1-distance) weighting, inverse distance weighting, and no weighting.

We anticipate a more accurate imputation technique will further improve the classification performance. Over the 12 datasets, kNN-CNI(SVM) and kNN-CNI(C4.5) each achieve about 2–4% higher average classification accuracies than

Table 2. Average Classification accuracy over the 12 datasets for instance-based algorithms using different nominal imputers as preprocessors (from top to bottom, bolded are best values) (a) kNN using (1-distance) weighting, (b) kNN using 1/*distance* weighting, (c) kNN without distance weighting

Missing Ratio(%)	Original kNN	kNN-MCI	kNN-CNI(C4.5)	kNN-CNI(SVM)	biggest-lift (%)
10	83.02	83.94	84.27	**84.55**	1.85
20	80.72	81.35	**83.37**	83.06	3.28
30	78.48	80.01	80.83	**81.36**	**3.67**
40	76.14	78.11	78.57	**78.92**	3.65
50	74.14	75.54	**76.00**	75.86	2.51
average	78.5	79.79	80.61	**80.75**	2.99
Missing Ratio(%)	Original kNN	kNN-MCI	kNN-CNI(C4.5)	kNN-CNI(SVM)	biggest-lift (%)
10	82.44	83.99	84.36	**84.55**	2.57
20	81.21	82.06	**83.32**	82.92	2.6
30	79.2	79.97	80.82	**81.37**	2.74
40	76.6	78.03	78.57	**78.93**	**3.04**
50	73.89	75.57	**76.02**	75.9	2.88
average	78.67	79.93	80.62	**80.73**	2.76
Missing Ratio(%)	Original kNN	kNN-MCI	kNN-CNI(C4.5)	kNN-CNI(SVM)	biggest-lift (%)
10	82.67	83.72	83.97	**84.78**	2.56
20	80.57	81.94	**83.13**	82.38	3.17
30	78.34	79.61	80.77	**81.39**	**3.9**
40	76.17	77.88	78.64	**78.90**	3.58
50	73.73	75.34	**75.72**	75.7	2.7
average	78.3	79.7	80.44	**80.63**	3.18

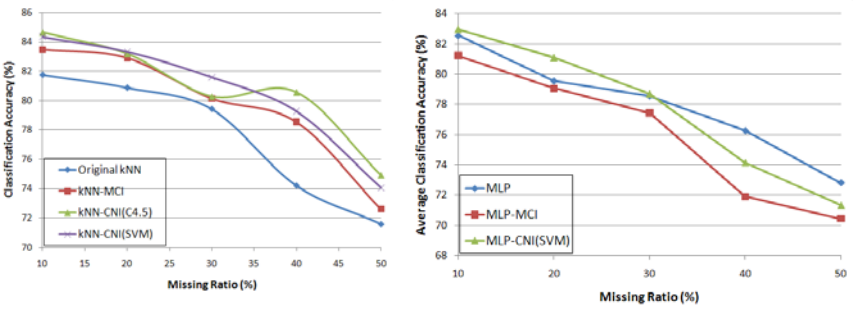


Fig. 4. (from left to right) (a) Case study on the dataset "Australian": using nominal imputers to preprocess incomplete training data helps improve the classification performance; and a better imputer (such as CNI(SVM)) will give a bigger lift of the classification accuracy. (b) The average classification accuracy of MLP, and MLP using MCI and CNI(SVM) as preprocessor for incomplete training data over all 12 datasets.

kNN on the original incomplete datasets, with 1-sided paired t-test $p < 0.0006$ and $p < 0.001$ respectively. KNN using MCI as the preprocessor slightly outperforms the original kNN by 1–2% on average, with $p < 0.01$.

3.3 The Impact of Nominal Imputers on Other Machine Learned Classifiers

We have shown that using high-quality imputation techniques to preprocess incomplete data will increase the classification performance for instance-based algorithms on nominal data. We now further investigate whether nominal imputers can help other classifiers perform better on incomplete nominal data, especially those with better missing data handling schemes – ie, where imputation may not be as critical.

We work on three machine learning algorithms, naïve Bayes [6], C4.5 [8], and MLP neural network [5], on the 12×5 datasets. We use one baseline imputer, MCI, and one top performer of our CNI imputers, CNI(SVM), as the nominal imputers for the training data before learning the classifiers. We compare with the classifiers that do not use imputation before training. Table 3 summarizes the results, and Figure 4(b) depicts the average accuracy for MLP, MLP-MCI, and MLP-CNI(SVM) on datasets of different missing ratios.

As Table 3 and Figure 4(b) indicate, accurate nominal imputers such as CNI(SVM) can help improve classification accuracy when the incomplete data has low missing ratios (e.g., at or below 20%), while using an inaccurate imputer such as MCI may lead to worse classification performance than the classifier that does not use any imputation for training data. When the missing ratio goes higher (i.e., missing ratio higher than 30%), neither CNI imputers nor MCI can help the learner to improve its classification accuracy. This is partially because many classifiers, such as NB, have more effective ways to deal with incomplete data than using imputation for highly sparse data. When the missing ratio is

Table 3. The average classification accuracies of the machine learned classifiers NB, C4.5, and MLP with and without using nominal imputers MCI and CNI(SVM)

	10.00%	20.00%	30.00%	40.00%	50.00%	average
NB	83.31	83.03	**81.85**	**80.65**	**77.82**	81.33
NB-MCI	82.73	81.87	81.07	79.07	75.43	80.03
NB-CNI(SVM)	**83.35**	**83.14**	81.62	79.15	76.69	80.79
C4.5	83.27	80.51	78.85	75.96	71.25	77.97
C4.5-MCI	83.31	**81.28**	**79.15**	**76.08**	**74.09**	78.78
C4.5-CNI(SVM)	**84.40**	81.11	77.56	75.34	71.77	78.04
MLP	82.55	79.56	78.56	**76.25**	**72.80**	77.95
MLP MCI	81.22	79.07	77.44	71.91	70.44	76.01
MLP -CNI(SVM)	**82.98**	**81.10**	**78.71**	74.17	71.34	77.66

high, the nominal imputers will become less accurate (see Section 3.1), and the benefit of using imputation will be offset by its inaccuracy.

Why are accurate nominal imputers effective for instance-based learning algorithms on both high-missing-ratio data and low-missing-ratio data? We suspect that this is partially due to the poor way that instance-based algorithms handle missing data, as using the maximized distance calculation can lead to poor classification accuracy, while using CNI imputation to preprocess incomplete nominal data before learning the base classifier will perform better.

As many real-world dataset are missing under 20% of the values, there is practical significance for our finding that an accurate nominal imputer (such as our proposed CNI imputation) can improve the classification performance of many machine learners on such incomplete datasets, even over learners that include effective built-in schemes for handling missing data.

4 Conclusions

Incomplete nominal data often make it difficult for learning algorithms to produce effective classifiers. Simple schemes of imputing the missing values, such as using the most common value for nominal missing data, are often not very effective. In this paper, we explore methods for improving classification performance for machine learning algorithms on incomplete nominal data. We first propose an easy-to-implement and effective nominal imputation algorithm: classifier-based nominal imputation (CNI), which fills in the missing values of attribute f_i by the values produced by a learned classifier that takes the other values in that instance, where this classifier is learned using as the training data the instances that contain the observed values of f_i. Our empirical results show that, of the 10 classification algorithms for imputation we investigated, the support vector machine (SVM) and C4.5 decision tree perform the best as CNI imputation learners. Our CNI imputers have significantly higher estimation accuracy than

the commonly used nominal imputation techniques, kNN imputation (kNNI) and most common imputation (MCI). Applying CNI imputers such as CNI(SVM) and CNI(C4.5) as the preprocessors for incomplete nominal training data can impressively improve the classification performance of instance-based learning algorithms on incomplete datasets over the entire range of missing ratios that we investigated. CNI imputers are also found helpful in improving the classification performance of machine learners that have effective missing data schemes, such as NB and C4.5, when the datasets have missing ratios at or below 20%.

In our future work, we plan to improve our CNI imputation algorithm by using the previously imputed values for the later iterations of imputations, in the order of imputations based on how dense or informative the attributes are. We also plan to explore ways to improve the instance-based learning algorithm using nominal imputations for other data types, such as mixed data (with both numeric and nominal data) and ordinal data.

Acknowledgement

This work was conducted when X. Su was with Varolii Corporation. R. Greiner was supported by the Alberta Ingenuity Centre for Machine Learning (AICML) and Canada's Natural Science and Engineering Research Council (NSERC). This work was supported by the Data Mining and Machine Learning Laboratory at Florida Atlantic University. We also thank Randall Wald for his help.

References

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine Learning 6, 37–66 (1991), http://dx.doi.org/10.1007/BF00153759, 10.1007/BF00153759
2. Batista, G., Monard, M.: An analysis of four missing data treatment methods for supervised learning. Applied Artificial Intelligence 17(5), 519–533 (2003)
3. Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
4. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11(1), 10–18 (2009)
5. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall PTR, Upper Saddle River (1998)
6. John, G., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, vol. 1, pp. 338–345. Citeseer (1995)
7. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization, pp. 185–208. MIT Press, Cambridge (1999), http://portal.acm.org/citation.cfm?id=299094.299105
8. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
9. Su, X., Khoshgoftaar, T., Greiner, R.: Using imputation techniques to help learn accurate classifiers. In: 20th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2008, vol. 1, pp. 437–444 (November 2008)

A Bayesian Framework for Learning Shared and Individual Subspaces from Multiple Data Sources

Sunil Kumar Gupta, Dinh Phung, Brett Adams, and Svetha Venkatesh

Department of Computing
Curtin University, Perth, Australia
sunil.gupta@postgrad.curtin.edu.au,
{d.phung,b.adams,s.venkatesh}@curtin.edu.au

Abstract. This paper presents a novel Bayesian formulation to exploit shared structures across multiple data sources, constructing foundations for effective mining and retrieval across disparate domains. We jointly analyze diverse data sources using a unifying piece of metadata (textual tags). We propose a method based on Bayesian Probabilistic Matrix Factorization (BPMF) which is able to explicitly model the partial knowledge common to the datasets using shared subspaces and the knowledge specific to each dataset using individual subspaces. For the proposed model, we derive an efficient algorithm for learning the joint factorization based on Gibbs sampling. The effectiveness of the model is demonstrated by social media retrieval tasks across single and multiple media. The proposed solution is applicable to a wider context, providing a formal framework suitable for exploiting individual as well as mutual knowledge present across heterogeneous data sources of many kinds.

1 Introduction

Recent developments in computing and information technology have enabled us to jointly analyze data from multiple sources such as multiple news feeds, social media streams and so on. Discovering structures and patterns from multiple data sources helps us to unravel certain commonalities and differences which otherwise is not possible when analyzing each data source separately. This information provides valuable inputs for various data mining and representation tasks. Whilst the data mining community has developed techniques to analyze a single data source, there is a need to develop formal frameworks for analyzing multiple data sources exploiting their common strengths.

However, modeling the data across multiple heterogeneous and disparate sources is a challenging task. For example, how do we model *text*, *image* and *video* together? At the semantic level, they provide much richer information together, and the question is how to exploit these strengths at lower levels? One solution is to exploit textual metadata for each data source in the form of *tags*. These tags are rich metadata sources, freely available across disparate data sources (images, videos, blogs etc.) and at topical or conceptual levels, they are often more meaningful than what current content processing methods extract [2]. However, tags can be ambiguous, incomplete and subjective [7] due to a lack of constraints during their creation. Due to this problem, performance of any data mining task using tags suffers significantly. Therefore, it is imperative to model the uncertainties of tag data to improve the performance of several data mining

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 136–147, 2011.

tasks. Work on tag denoising has been, broadly speaking, aimed at determining tag relevance through modification or the recommendation of additional tags [11,7]. But these approaches typically focus solely within the internal structure of a given tagging source, and are thus bounded by the information content and noise characteristics of the tagging source. Moreover, these methods, working individually for each data source, lack in exploiting the collective strengths of all data sources.

Addressing the problem of constructing a unified framework for disparate sources, we develop a Bayesian framework which can model the uncertainties of multiple data sources jointly using the textual tags from each source as a unified piece of metadata. Our method allows multiple data sources to exploit collective strength by learning probabilistic shared subspaces and, at the same time, *crucially* retains the differences of each data source by learning probabilistic individual subspaces. Retaining the differences between data sources is very important; ignoring this aspect often leads to "negative knowledge transfer". Another strength of the proposed framework is that both the shared and individual subspaces are probabilistic in nature, which helps in modeling the uncertainties involved in real world applications. Similar work by Gupta et al [5] also models the shared and individual subspaces but has certain limitations. First, their framework is restrictive in part as it can model only nonnegative data sources. Second, their model supports only two data sources, which renders the model unusable when working with more than two data sources. Third, the subspaces learnt are not probabilistic and do not cater for uncertainties such as missing tags and tag ambiguity.

Previous works on shared subspace learning are mainly focused on supervised or semi-supervised learning. Ji et al [6] and Yan et al [12] provide frameworks for extracting shared structures in multi-label classification by learning a common subspace which is assumed to be shared among multiple labels. Si et al [10] propose a family of transfer subspace learning algorithms by minimizing Bregman divergence between the distributions of the training and test samples. This approach, while being fairly generic for transfer learning, is not appropriate for multi-task learning and can not exploit the knowledge strengths from the multiple data sources. In another work, Gu and Zhou [4] propose multi-task clustering by learning a shared subspace for all tasks. This approach provides a way to learn shared subspaces, but has no way of controlling the sharing level, a crucial aspect when dealing with heterogeneous data sources. Moreover, the sharing is imposed among all tasks which is unrealistic in many scenarios.

Our framework is based on the state-of-the-art Bayesian probabilistic matrix factorization (BPMF) model, recently proposed in [9]. We extend BPMF to enable joint modeling of multiple data sources deriving common and individual subspaces, and derive inference algorithms using Rao-Blackwellized Gibbs Sampling (RBGS). To demonstrate the usefulness of our approach, we examine two applications – improving social media retrieval using auxiliary sources, and cross-social media retrieval. We use three disparate data sources – Flickr, YouTube and Blogspot – to show the effectiveness of shared subspace learning frameworks.

Our main contributions are :

- The construction of a novel Bayesian shared subspace learning framework for extraction of shared and individual subspaces across an arbitrary number of data sources using joint matrix factorization.

- Efficient inference based on RBGS (Gibbs sampling) for joint factorization.
- Algorithms for retrieval within one social medium or across disparate social media using Bayesian shared subspace learning framework.
- Two real-world applications using three popular social media sources: Blogspot, Flickr and YouTube. We demonstrate (1) improvement in social media retrieval by leveraging auxiliary media, and (2) effective cross-social media retrieval.

The novelty of our approach lies in the framework and algorithms for learning *across* diverse data sources. Our probabilistic shared and individual subspaces not only exploit the mutual strengths of multiple data sources but also handle the uncertainties.

The significance of our work lies in the fact that theoretical extensions made to BPMF [9] for multiple data sources allow for the flexible transfer of knowledge across disparate media. In addition, RBGS sampling derived for our model achieves better inference (i.e. Markov chain mixes better) compared to [9]. Our work brings a broad scope of open opportunities and applications—it is appropriate wherever one needs to exploit multiple and heterogeneous datasets for knowledge transfer among them, such as collaborative filtering or sentiment analysis.

The rest of the paper is organized as follows. Section 2 presents the Bayesian shared subspace learning formulation and describes Gibbs sampling based inference procedure. Section 3 and 4 demonstrate the applicability of the proposed framework to social media retrieval applications. Conclusions are drawn in Section 5.

2 Bayesian Shared Subspace Learning (BSSL)

We introduce a framework for learning individual and shared subspaces across an arbitrary number of data sources. Let a set of n data sources be represented by data matrices $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$, which, for example, can be term-document matrices (each row a word and each column a document with *tf-idf* features [1]) for retrieval applications or user rating matrices (each row a user and each column an item with ratings as features) for collaborative filtering applications. We assume that matrices $\boldsymbol{X}_i$ have the same number of rows. Whenever it is not the case, one can always merge the vocabularies of each data source to form a common vocabulary. Our goal is to factorize each matrix $\boldsymbol{X}_i$ as $\boldsymbol{X}_i = \boldsymbol{W}_i \cdot \mathbf{H}_i + \mathbf{E}_i$ such that the decomposition captures arbitrary *sharing* of basis vectors among data sources whilst preserving their *individual* bases. For example, when $n = 2$, we create three subspaces: a shared subspace matrix W_{12} and two individual subspaces W_1, W_2. We thus write $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ as

$$\boldsymbol{X}_1 = \underbrace{[W_{12} \mid W_1]}_{\boldsymbol{W}_1} \underbrace{\begin{bmatrix} H_{1,12} \\ \\ H_{1,1} \end{bmatrix}}_{\mathbf{H}_1} + \mathbf{E}_1 \text{ and } \boldsymbol{X}_2 = \underbrace{[W_{12} \mid W_2]}_{\boldsymbol{W}_2} \underbrace{\begin{bmatrix} H_{2,12} \\ \\ H_{2,2} \end{bmatrix}}_{\mathbf{H}_2} + \mathbf{E}_2 \qquad (1)$$

To define subspaces at data source level, we define $\boldsymbol{W}_1 = [W_{12} \mid W_1]$ and $\boldsymbol{W}_2 = [W_{12} \mid W_2]$ for the two data sources. Note however that the encoding coefficients corresponding to the shared subspace W_{12} are different, and thus, an extra subscript is used to make it explicit in $H_{1,12}$ and $H_{2,12}$. *Notation-wise, we use bold symbols* $\boldsymbol{W}$,$\mathbf{H}$ *to*

denote the entire decomposition at a data source level and normal capital letters W, H at the shared level. $\mathbf{E}_1$ and $\mathbf{E}_2$ denote the residual factorization error.

To see how we can generalize these expressions for n datasets, we continue with this example by constructing the power set over $\{1,2\}$ as $S(2) = \{\emptyset, \{1\}, \{2\}, \{1,2\}\}$. Our intention is to create an index set over the subscripts $\{1,2,12\}$ used in matrices presented in Eq (1) so that a summation can be conveniently written. To do so, we further use $S(2,i)$ to denote the subset of $S(2)$ in which only elements involving i are retained, i.e. $S(2,1) = \{\{1\},\{1,2\}\}$ and $S(2,2) = \{\{2\},\{1,2\}\}$. With a slight relaxation in the set notation, we rewrite them as $S(2,1) = \{1,12\}$ and $S(2,2) = \{2,12\}$. Thus, Eq (1) can be re-written as

$$\boldsymbol{X}_1 = \sum_{v\in\{1,12\}} W_v \cdot H_{1,v} + \mathbf{E}_1 \text{ and } \boldsymbol{X}_2 = \sum_{v\in\{2,12\}} W_v \cdot H_{2,v} + \mathbf{E}_2$$

For a set of n datasets, let $S(n)$ denote the set of all subsets over $\{1,2,\ldots n\}$ and for each $i = 1,\ldots,n$, denote by $S(n,i) = \{v \in S(n) \mid i \in v\}$ the index set associated with the i-th data source. Our proposed shared subspace learning framework seeks a set of expression in the following forms for $i = 1,\ldots,n$

$$\boldsymbol{X}_i = \boldsymbol{W}_i \cdot \mathbf{H}_i + \mathbf{E}_i = \sum_{v\in S(n,i)} W_v \cdot H_{i,v} + \mathbf{E}_i,\ i = 1,\ldots,n \tag{2}$$

It is also clear from Eq (2) that the total subspace $\boldsymbol{W}_i$ and its corresponding encoding matrix $\mathbf{H}_i$ for the i-th data matrix are horizontally augmented matrices over all W_v and vertically augmented over all $H_{i,v}$ for $v \in S(n,i)$ respectively. That is, if we explicitly list the elements of $S(n,i)$ as $S(n,i) = \{v_1, v_2, \ldots, v_Z\}$ then $\boldsymbol{W}_i$, $\mathbf{H}_i$ are

$$\boldsymbol{W}_i = [W_{v_1} \mid W_{v_2} \mid \ldots \mid W_{v_Z}] \text{ and } \mathbf{H}_i = \begin{bmatrix} H_{i,v_1} \\ \vdots \\ H_{i,v_Z} \end{bmatrix} \tag{3}$$

2.1 Bayesian Representation

We treat the residual errors ($\mathbf{E}_i$) probabilistically and model each $\mathbf{E}_i$, $\forall i$ as i.i.d. and normally distributed with mean zero and precisions $\Lambda_{\boldsymbol{X}_i}$. Although we consider Bayesian shared subspace learning for arbitrary number of data sources, for simplicity, we show the graphical model for the case of two data sources in Figure 1. For each $i \in \{1,2,\ldots n\}$ and $v \in S(n,i)$, our probabilistic model is then given as

$$p(\boldsymbol{X}_i(m,l) \mid \boldsymbol{W}_i, \mathbf{H}_i, \Lambda_{\boldsymbol{X}_i}) = \mathcal{N}\left(\boldsymbol{X}_i(m,l) \mid \boldsymbol{W}_i^{(m)} \mathbf{H}_i^{[l]}, \Lambda_{\boldsymbol{X}_i}^{-1}\right)$$

$$p(W_v \mid \mu_{W_v}, \Lambda_{W_v}) = \prod_{m=1}^{M} \mathcal{N}\left(W_v^{(m)} \mid \mu_{W_v}, \Lambda_{W_v}^{-1}\right)$$

$$p(\mathbf{H}_i \mid \mu_{\mathbf{H}_i}, \Lambda_{\mathbf{H}_i}) = \prod_{l=1}^{N_i} \mathcal{N}\left(\mathbf{H}_i^{[l]} \mid \mu_{\mathbf{H}_i}, \Lambda_{\mathbf{H}_i}^{-1}\right)$$

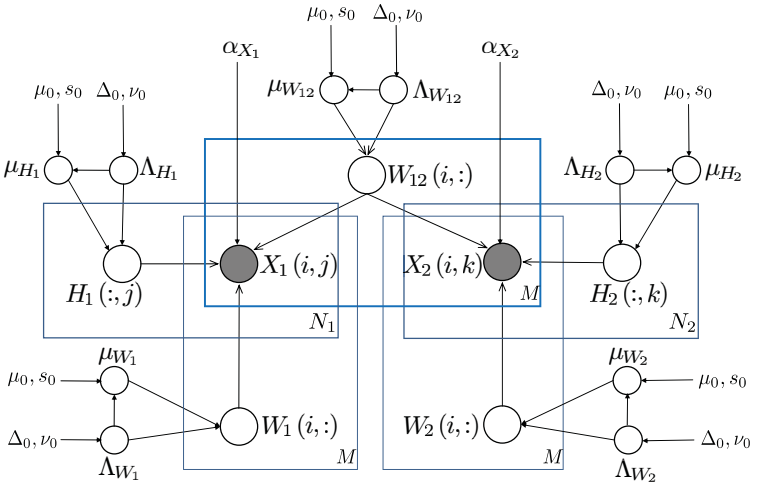


Fig. 1. Graphical Model for BSSL (a special case for two data sources, i.e. $n = 2$)

where $\mathbf{B}^{(m)}$ *denotes the* m*-th row, i.e.* $\mathbf{B}(m,:)$ *while* $\mathbf{B}^{[l]}$ *denotes the* l*-th column, i.e.* $\mathbf{B}(:,l)$ *of a matrix* $\mathbf{B}$. Since $\mathbf{E}_i$'s are i.i.d., we set $\Lambda_{\boldsymbol{X}_i} = \alpha_{\boldsymbol{X}_i}\mathbf{I}$ for each i. Going fully Bayesian, we further use a normal-Wishart prior on the parameters $\{\mu_{W_v}, \Lambda_{W_v}\}$. The normal-Wishart prior is given by

$$p(\mu_{W_v}, \Lambda_{W_v} \mid \Psi_0) = \mathcal{N}\left(\mu_{W_v} \mid \mu_0, (s_0\Lambda_{W_v})^{-1}\right)\mathcal{W}(\Lambda_{W_v} \mid \Delta_0, \nu_0)$$

where $\mathcal{W}(. \mid \Delta_0, \nu_0)$ is Wishart distribution with $K_v \times K_v$ scale matrix Δ_0 and ν_0 degree of freedom. Similar priors are placed on the parameters $\{\mu_{\mathbf{H}_i}, \Lambda_{\mathbf{H}_i}\}$. For future reference, we define all the hyperparameters as $\Psi_0 \triangleq \{\mu_0, s_0, \Delta_0, \nu_0, \alpha_{\boldsymbol{X}_1}, \ldots, \alpha_{\boldsymbol{X}_n}\}$.

2.2 Gibbs Inference

Given data matrices $\{\boldsymbol{X}_i\}_{i=1}^n$, the goal of BSSL is to learn the factor matrices W_v and $\mathbf{H}_i$ for all $i \in \{1, 2, \ldots n\}$ and $v \in S(n, i)$. In our Bayesian setting, this translates to performing posterior inference on the distribution of random (row or column) vectors from W_v and $\mathbf{H}_i$. Since we are treating these vectors as Gaussian with proper conjugate prior normal-Wishart distribution, posterior inference can be conveniently carried out using Gibbs sampling, which is guaranteed to converge asymptotically.

In a typical Gibbs sampling setting, our state space for sampling is $\{W_v, \mu_{W_v}, \Lambda_{W_v}\}$ and $\{\mathbf{H}_i, \mu_{\mathbf{H}_i}, \Lambda_{\mathbf{H}_i}\}$ conditioned on the hyperparameters Ψ_0 and data $\{\boldsymbol{X}_i\}_{i=1}^n$. However, $\{\mu_{W_v}, \Lambda_{W_v}\}$ and $\{\mu_{\mathbf{H}_i}, \Lambda_{\mathbf{H}_i}\}$ are nuisance parameters which can be integrated out to reduce the variance of the Gibbs samples (for better mixing of Markov chain) – a scheme which is known as Rao-Blackwellized Gibbs Sampling (RBGS).

After integrating out these nuisance parameters, our state space reduces to only the factor matrices $\{W_v, \mathbf{H}_i\}$ for all $i \in \{1, 2, \ldots n\}$ and $v \in S(n, i)$. Our Gibbs sampler then iteratively samples each row of W_v and column of $\mathbf{H}_i$ conditioned on the observed

data and the remaining set of variables in the state space from the previous Gibbs iteration. Algorithm 1 outlines these sampling steps while the rest of this section shall briefly explain how to obtain the Gibbs conditional distributions as in Eqs (6-9).

Recalling $\boldsymbol{W}_i$ and $\mathbf{H}_i$ from Eq. (3), the conditional distribution over $W_v^{(m)}$, conditioned on matrices W_u for all $u \in S(n,i)$ except v, all the rows of matrix W_v except m-th row (denoted by $W_v^{(\backslash m)}$), the coefficient matrices $\mathbf{H}_i$ for all i, observed data $\boldsymbol{X}_i$ for all i and the hyperparameters Ψ_0, is given by

$$p\left(W_v^{(m)} \mid \boldsymbol{X}_{1:n}, W_v^{(\backslash m)}, W_{\{u\in S(n,i)\}\backslash v}, \mathbf{H}_{1:n}, \Psi_0\right) \propto \left[\prod_{i=1}^{n}\prod_{l=1}^{N_i} p\left(\boldsymbol{X}_i(m,l) \mid \boldsymbol{W}_i^{(m)}\mathbf{H}_i^{[l]}, \Lambda_{\boldsymbol{X}_i}^{-1}\right)\right] \times p\left(W_v^{(m)} \mid W_v^{(\backslash m)}, \Psi_0\right) \quad (4)$$

Algorithm 1. Rao-Blackwellized Gibbs Sampling (RBGS) for BSSL.

1: **Input**: Hyperparameters Ψ_0, number of samples L.
2: For each i, initialize matrices $\boldsymbol{W}_i$, $\mathbf{H}_i$ randomly.
3: **for** $r = 1, \ldots, L$ **do**
4: For each v, draw r-th sample $[W_v]^r$ from normal distribution parameterized by Eqs.(6-7).
5: For each i, draw r-th sample $[\mathbf{H}_i]^r$ from normal distribution parameterized by Eqs.(8-9).
6: **end for**
7: For each v and i, get an estimate of W_v and $\mathbf{H}_i$ using the Gibbs samples as $W_v \approxeq \frac{1}{L}\sum_{r=1}^{L}[W_v]^r$, $\mathbf{H}_i \approxeq \frac{1}{L}\sum_{r=1}^{L}[\mathbf{H}_i]^r$.
8: **Output**: Samples $\{[W_v]^r\}_{r=1}^{L}$, $\{[\mathbf{H}_i]^r\}_{r=1}^{L}$ and estimates W_v, $\mathbf{H}_i$ for each v and i.

Note that the above posterior is proportional to the data-likelihood as a function of $W_v^{(m)}$ and $\mathbf{H}_i$ for each i and the predictive distribution of $W_v^{(m)}$ given $W_v^{(\backslash m)}$. The predictive distribution of $W_v^{(m)}$ conditioned on $W_v^{(\backslash m)}$ and Ψ_0 is obtained by integrating over the parameters of the normal–inverse–Wishart posterior distribution and is *multivariate Student–t* [3]. Assuming $\nu_l > K_v + 1$, this predictive density has finite covariance and is known to be approximated well by a normal distribution through matching the first two moments [3]. Thus, the predictive distribution is given as

$$p\left(W_v^{(m)} \mid W_v^{(\backslash m)}, \Psi_0\right) \approx \mathcal{N}\left(W_v^{(m)} \mid \mu_{W_v^{(m)}}^{pred}, \Lambda_{W_v^{(m)}}^{pred}\right) \quad (5)$$

$$\text{where} \quad \mu_{W_v^{(m)}}^{pred} = \frac{s_0\mu_0 + \sum_{\substack{l=1\\ l\neq m}}^{M} W_v^{(l)}}{s_0 + (M-1)}, \; \Lambda_{W_v^{(m)}}^{pred} = \frac{(s_m+1)\,\Delta_m^{-1}}{s_m\,(\nu_m - K_{W_v} - 1)},$$

$$\Delta_m^{-1} = \Delta_0^{-1} + \sum_{\substack{l=1\\ l\neq m}}^{M} W_v^{(l)}\left(W_v^{(l)}\right)^{\mathsf{T}} + \frac{s_0\,(M-1)}{s_0 + M - 1}\left(\mu_0 - \bar{\mu}_{W_v\backslash m}\right)\left(\mu_0 - \bar{\mu}_{W_v\backslash m}\right)^{\mathsf{T}},$$

$$\bar{\mu}_{W_v \setminus m} \triangleq \frac{1}{(M-1)} \sum_{\substack{l=1 \\ l \neq m}}^{M} W_v^{(l)},\ s_m = s_0 + M - 1,\ \nu_m = \nu_0 + M - 1,\ W_v \in \mathbb{R}^{M \times K_v}$$

Using Eqs (4) and (5), the posterior distribution can be written as

$$p\left(W_v^{(m)} \mid \boldsymbol{X}_{1:n}, W_v^{(\setminus m)}, W_{\{u:u \neq v\}}, \mathbf{H}_{1:n}, \Psi_0\right) = \mathcal{N}\left(W_v^{(m)} \mid, \mu_{W_v^{(m)}}^{post}, \Lambda_{W_v^{(m)}}^{post}\right) where$$

$$\Lambda_{W_v^{(m)}}^{post} = \Lambda_{W_v^{(m)}}^{pred} + \sum_{i=1}^{n} H_{i,v} \Lambda_{\boldsymbol{X}_i} H_{i,v}^{\mathsf{T}} \tag{6}$$

$$\left(\mu_{W_v^{(m)}}^{post}\right)^{\mathsf{T}} = \left[\Lambda_{W_v^{(m)}}^{post}\right]^{-1} \left[\Lambda_{W_v^{(m)}}^{pred} \left(\mu_{W_v^{(m)}}^{pred}\right)^{\mathsf{T}} + \sum_{i=1}^{n} H_{i,v} \Lambda_{\boldsymbol{X}_i} \left(\mathbf{A}_{i,v}^{(m)}\right)^{\mathsf{T}}\right] \tag{7}$$

and $\mathbf{A}_{i,v}^{(m)} \triangleq \boldsymbol{X}_i^{(m)} - \sum_{\{u:u \neq v\}} W_u^{(m)} H_{i,u}$. Similar to $W_v^{(m)}$, the posterior distribution over the l-th column of matrix $\mathbf{H}_i$ conditioned on its remaining columns is normally distributed with mean vector and precision matrix as

$$\Lambda_{\mathbf{H}_i^{[l]}}^{post} = \Lambda_{\mathbf{H}_i^{[l]}}^{pred} + \boldsymbol{W}_i^{\mathsf{T}} \Lambda_{\boldsymbol{X}_i} \boldsymbol{W}_i \tag{8}$$

$$\mu_{\mathbf{H}_i^{[l]}}^{post} = \left[\Lambda_{\mathbf{H}_i^{[l]}}^{post}\right]^{-1} \left[\Lambda_{\mathbf{H}_i^{[l]}}^{pred} \mu_{\mathbf{H}_i^{[l]}}^{pred} + \boldsymbol{W}_i^{\mathsf{T}} \Lambda_{\boldsymbol{X}_i} \boldsymbol{X}_i^{[l]}\right] \tag{9}$$

2.3 Subspace Dimensionality and Complexity Analysis

Let the number of rows in $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ be M and the number of columns be N_i, giving $M \times N_i$ dimension for $\boldsymbol{X}_i$. Since each $\boldsymbol{W}_i$ consists of an augmentation of individual and shared subspaces W_v, we use K_v to denote the number of basis vectors in W_v. Assuming R_i to be the total number of basis vectors in $\boldsymbol{W}_i$, we have $\sum_{v \in S(n,i)} K_v = R_i$. Determining the value of K_v is a model selection problem and depends upon the common features among various data sources. According to a heuristic proposed in [8], a rule of thumb is to use $K_v \approx \sqrt{M_v/2}$ where M_v is the number of common features in sharing configuration v. Following the above notation for the dimensionalities of matrices, for each $i \in \{1, 2, \ldots n\}$, assuming $R_i < M$ (generally the case for real world data), the complexity of sampling $\mathbf{H}_i$ matrices from its posterior distributions is $\mathcal{O}(M \times N_i \times R_i)$ whereas the complexity involved in sampling $\boldsymbol{W}_i$ matrices from its posterior distributions is $\mathcal{O}\left(M \times N_i \times R_i^2\right)$. Thus the computation complexity of BSSL remains similar to BPMF model [9] and does not grow any further.

3 Social Media Applications

We demonstrate the usefulness of BSSL in two real world applications. (1) Improving social media retrieval in one medium by transferring knowledge from auxiliary media sources. (2) Performing retrieval across multiple media. The first application can be

seen as an multi-task learning application, whereas the second application is a direct manifestation of mining from multiple data sources.

3.1 BSSL Based Social Media Retrieval

Let the tag-item matrix of the target medium (from which retrieval is to be performed) be denoted as $\boldsymbol{X}_k$. Further, let us assume that we have many other auxiliary media sources which share some features with the target medium. Let the tag-item matrices of these auxiliary media be denoted by $\boldsymbol{X}_j$, $j \neq k$. In a multi-task learning setting, we leverage these auxiliary sources to improve the retrieval precision for the target medium, and given a set of query keywords S_Q, a vector q of length M (vocabulary size) is constructed by putting *tf-idf* values at each index where the vocabulary contains a word from the keywords set or else setting it to zero. Next, we follow Algorithm 2 for BSSL based retrieval.

3.2 BSSL Based Cross-Social Media Retrieval

To retrieve items across media, we use the common subspace among them along with the corresponding coefficient matrices for each medium. As an example, for $n = 3$ (three media sources), we use the common subspace matrix W_{123} and coefficient matrices $H_{1,123}$, $H_{2,123}$ and $H_{3,123}$ for first, second and third medium respectively.

Similar to subsection 3.1, we construct a vector q of length M using a set of query keywords S_Q. We proceed similar to Algorithm 2 with the following differences. Given q, we wish to retrieve relevant items from each domain, which is performed by projecting q onto the augmented common subspace matrix (W_{123} for the case when $n = 3$ media sources) to get its representation h in the common subspace. Next, we compute similarity between h and the columns of matrices $H_{1,123}$, $H_{2,123}$ and $H_{3,123}$ (the representation of media items in the common subspace spanned by columns of W_{123}) to find similar items from medium 1, 2 and 3 respectively. The results are ranked based on these similarity scores either individually or jointly.

For both retrieval applications, we use *cosine-similarity* as it seems to be more robust than Euclidean distance based similarity measures in high-dimensional spaces. As we are dealing with distributions, we also tried out *KL-divergence* based similarity measures, but *cosine-similarity* gives better results.

Algorithm 2. Social Media Retrieval using BSSL.

1: **Input**: Target $\boldsymbol{X}_j$, auxiliaries $\boldsymbol{X}_k$, $k \neq j$, query q , set of items from medium j, denoted as $\mathcal{I} = \{I_1, I_2, ..., I_{N_j}\}$, number of items to be retrieved N.
2: Get Gibbs estimates of $\boldsymbol{W}_j$ and $\mathbf{H}_j$ using Algorithm 1.
3: Project q onto the subspace spanned by $\boldsymbol{W}_j$ to get h as $h = \boldsymbol{W}_j^{\dagger} q$ where † is Moore-Penrose pseudoinverse of a matrix.
4: For each item (indexed by m) in $\boldsymbol{X}_j$, with its subspace representation $h_m = m$-th column of $\mathbf{H}_j$, compute its cosine similarity with query projection h : $\text{sim}\,(h, h_m) = \frac{h^{\top} h_m}{\|h\|_2 \|h_m\|_2}$
5: **Output**: Return the top N items in decreasing order of similarity.

Table 1. Description of the YouTube-Flickr-Blogspot data set

Media	*Dataset Size*	*Concepts Used for Creating Dataset*	*Avg. Tags/Item* (rounded)
Blogspot	10000	'Academy Awards', 'Australian Open', 'Olympic Games', 'US Election', 'Cricket World Cup', 'Christmas', 'Earthquake'	6
Flickr	20000	'Academy Awards', 'Australian Open', 'Olympic Games', 'US Election', 'Holi', 'Terror Attacks', 'Christmas'	8
YouTube	7000	'Academy Awards', 'Australian Open', 'Olympic Games', 'US Election', 'Global Warming', 'Terror Attacks', 'Earthquake'	7

4 Experiments

4.1 Dataset

To conduct the experiments, we created a social media dataset using three different sources : YouTube[1], Flickr[2] and Blogspot[3]. To obtain the data, we first chose common concepts – 'Academy Awards', 'Australian Open', 'Olympic Games', 'US Election' – and queried all three websites using their service APIs. To have some pairwise sharing, we additionally used the concept 'Christmas' to query Blogspot and Flickr, 'Terror Attacks' to query YouTube and Flickr, and 'Earthquake' to query Blogspot and YouTube. Lastly, to retain some differences between each medium, we used the concepts 'Cricket World Cup', 'Holi' and 'Global Warming' to query Blogspot.com, Flickr and YouTube respectively. Table 1 provides further details of the dataset size and average tag counts for each data source. The total number of tags from all three sources combined is 3740.

4.2 Subspace Learning and Parameter Setting

For clarity, let us denote YouTube, Flickr and Blogspot tag-item matrices as $\boldsymbol{X}_1$, $\boldsymbol{X}_2$ and $\boldsymbol{X}_3$ respectively. To learn BSSL based factorization, we use Eqs (6)–(9) to sample W and H matrices. Recalling the notation K_v (dimensionality of subspace spanned by W_v), for learning factorization, we set the individual subspace dimensions as $K_1 = K_2 = K_3 = 10$, pairwise shared subspace dimensions as $K_{12} = K_{23} = K_{13} = 15$, and the common to all subspace dimension as $K_{123} = 25$. To obtain these parameters, we first initialize them using the heuristic described in [8] and then do cross-validation based on retrieval precision. In addition, we also set the error precisions $\alpha_{\boldsymbol{X}_1} = \alpha_{\boldsymbol{X}_2} = \alpha_{\boldsymbol{X}_3} = 2$, hyperparameters $\mu_0 = [0, \ldots, 0]^{\mathsf{T}}$, $s_0 = 1$, $\Delta_0 = \mathbf{I}$ and $\nu_0 = K_v$ for corresponding W_v, $H_{i,v}$. The values of $\alpha_{\boldsymbol{X}_i}$ depend upon the quality of the tags and a small value implies high tag uncertainty. For the dataset described

[1] http://code.google.com/apis/youtube/overview.html
[2] http://www.flickr.com/services/api/
[3] http://www.blogger.com/

above, Gibbs sampling usually takes around 50 iterations to converge (convergence plots are omitted due to space restrictions), however, we collect 100 samples to ensure convergence. The first 20 samples are rejected for "burn-in" and the remaining 80 Gibbs samples are averaged to get an estimate of W_v, $H_{i,v}$ matrices.

4.3 Experiment 1: Social Media Retrieval Using Auxiliary Sources

To carry out our experiments, we choose YouTube as the target dataset and Blogspot and Flickr as auxiliary datasets. To perform BSSL based retrieval from YouTube, we first generate samples of basis matrix $\boldsymbol{W}_1 \triangleq [W_1 \mid W_{12} \mid W_{13} \mid W_{123}]$ and representation matrix $\mathbf{H}_1 \triangleq [H_{1,1} \mid H_{1,12} \mid H_{1,13} \mid H_{1,123}]$ according to Eqs (6)–(9) and then get an estimate of $\boldsymbol{W}_1$ and $\mathbf{H}_1$ following Algorithm 2.

To compare the performance with other methods, we choose three baselines. The first baseline performs retrieval by matching the tag-lists of videos without any subspace learning. To get the similarity with other items, Jaccard coefficent[4] is used and the results are ranked based on the similarity scores. The second baseline is retrieval work based on subspace learning using Principle Component Analysis (PCA). For the third baseline, we use a recent state-of-the-art BPMF model proposed in [9] for Bayesian matrix factorization. For both second and third baselines, we do not use any auxiliary data (e.g. tags of Flickr or Blogspot) and use the tags of YouTube only, but increase the YouTube datasize so as to keep the total datasize equal to the combined data (target + auxiliary) used for the first baseline to make the comparison fair.

To evaluate our retrieval algorithm, we use a query set of 20 concepts defined as $\mathbb{Q} =$ {'beach', 'america', 'bomb', 'animal', 'bank', 'movie', 'river', 'cable', 'climate', 'federer', 'disaster', 'elephant', 'europe', 'fire', 'festival', 'ice', 'obama', 'phone', 'santa', 'tsunami'}. Since there is no public groundtruth available, we manually go through the set of retrieved items and evaluate the results.

Figure 2 compares the retrieval performance of BSSL with all the baselines in terms of *precision-scope (P@N)* curve[5], *mean average precision (MAP)* and *11-point precision-recall curve* [1]. Figure 2 clearly shows that BSSL outperforms the baselines in terms of all three evaluation criteria. Although, BPMF performs better than PCA due to its ability to handle uncertainties well, it can not surpass BSSL as it is confined to the tag data of YouTube only. Intuitively, BSSL is able to utilize the related data from auxiliary sources and resolve the tag ambiguities by reducing the subjectivity and incompleteness of YouTube tags. In essence, the use of multiple sources in subspace learning helps discover improved tag co-occurrences and gives better results.

4.4 Experiment 2: Cross Media Retrieval

The effectiveness of BSSL for cross-media retrieval is demonstrated using the YouTube-Flickr-Blogspot dataset with the subspace learning parameters as in subsection 4.2. For evaluation, we again use *precision-scope (P@N), mean average precision (MAP)* and *11-point interpolated precision-recall curves.* Let q be a query term, G_i be the ground

[4] Jaccard $(A, B) = |A \cap B| \,/\, |A \cup B|$.

[5] For example, P@10 is the retrieval precision when considering the top 10 retrieved items.

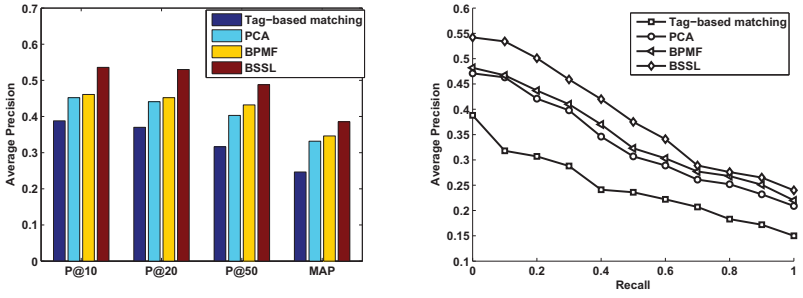


Fig. 2. YouTube retrieval results using auxiliary sources Flickr and Blogspot (a) Precision-Scope, MAP (b) 11-point interpolated Precision-Recall; for tag-based matching (baseline 1), PCA (baseline 2), BPMF [9] (baseline 3) and proposed BSSL

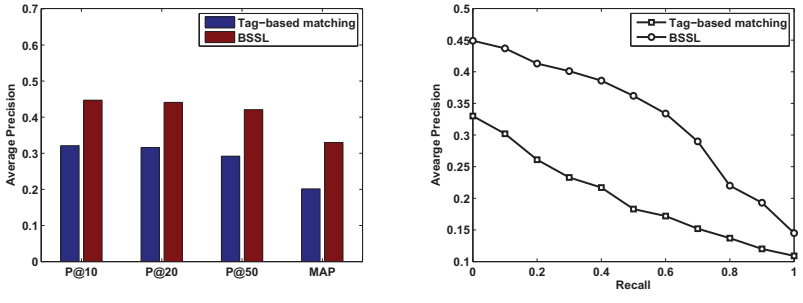


Fig. 3. Cross-media retrieval results: (a) precision-scope, MAP (b) 11-point interpolated precision-recall; for tag-based matching and proposed BSSL

truth set for the i-th medium and A_i be the answer set for query q using a retrieval method for the i-th medium. Then, the precision and recall measures for cross-media retrieval are

$$Precision = \frac{\sum_{i=1}^{n} |A_i \cap G_i|}{\sum_{i=1}^{n} |A_i|}, \quad Recall = \frac{\sum_{i=1}^{n} |A_i \cap G_i|}{\sum_{i=1}^{n} |G_i|}$$

As far as baselines are concerned, we note that both BPMF and PCA are not applicable for cross-media retrieval as they do not support analysis of multiple data sources in their standard form. Therefore, we compare the performance of BSSL against tag-based matching (based on Jaccard coefficient without any subspace learning) only. Other works on cross-media retrieval [13,14] use the concept of a Multimedia Document (MMD), which requires *co-occurring* multimedia objects on the same webpage which is not available in our case. Therefore, these methods can not be applied directly.

Figure 3 depicts the cross-media retrieval results across all three media - Blogspot, Flickr and YouTube. To generate the graphs, we again average the retrieval results over

the query set $\mathbb{Q}$ defined in subsection 4.3. It can be seen from Figure 3 that BSSL significantly outperforms tag based matching in terms of all three evaluation criteria. Improvement in terms of *MAP* criteria is around 13%. This improvement in performance is due to the learning of shared subspaces which not only handle the problem of 'synonymy' and 'polysemy' in tag-space, but also the uncertainties probabilistically.

5 Conclusion

We have presented a Bayesian framework to learn individual and shared subspaces from multiple data sources (BSSL) and demonstrated its application to social media retrieval across single and multiple media. Our framework, being based on the principle of Bayesian probabilistic matrix factorization (BPMF) [9], provides an efficient algorithm to learn the subspace. Our Gibbs sampler (RBGS) provides better Markov chain mixing than BPMF without increasing the complexity of the model. Our experiments have demonstrated that BSSL significantly outperforms the baseline methods for both retrieval tasks on Blogspot, Flickr and YouTube datasets. More importantly, our solution provides a generic framework to exploit collective strengths from heterogeneous data sources and we foresee its wider adoption in cross-domain data mining and beyond.

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern information retrieval. Addison-Wesley, Reading (1999)
2. Datta, R., Joshi, D., Li, J., Wang, J.: Image retrieval: ideas, influences, and trends of the new age. ACM Computing Surveys (CSUR) 40(2), 1–60 (2008)
3. Gelman, A.: Bayesian data analysis. CRC Press, Boca Raton (2004)
4. Gu, Q., Zhou, J.: Learning the shared subspace for multi-task clustering and transductive transfer classification. In: ICDM, pp. 159–168 (2009)
5. Gupta, S.K., Phung, D., Adams, B., Tran, T., Venkatesh, S.: Nonnegative shared subspace learning and its application to social media retrieval. In: SIGKDD, pp. 1169–1178 (2010)
6. Ji, S., Tang, L., Yu, S., Ye, J.: Extracting shared subspace for multi-label classification. SIGKDD, 381–389 (2008)
7. Li, X., Snoek, C.G.M., Worring, M.: Learning social tag relevance by neighbor voting. IEEE Transactions on Multimedia 11(7), 1310–1322 (2009)
8. Mardia, K.V., Bibby, J.M., Kent, J.T.: Multivariate analysis. Academic Press, NY (1979)
9. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using markov chain monte carlo. In: ICML, pp. 880–887 (2008)
10. Si, S., Tao, D., Geng, B.: Bregman divergence based regularization for transfer subspace learning. IEEE Transactions on Knowledge and Data Engineering 22(7), 929–942 (2009)
11. Sigurbjörnsson, B., Van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: WWW, pp. 327–336 (2008)
12. Yan, R., Tesic, J., Smith, J.: Model-shared subspace boosting for multi-label classification. SIGKDD, 834–843 (2007)
13. Yang, Y., Xu, D., Nie, F., Luo, J., Zhuang, Y.: Ranking with local regression and global alignment for cross media retrieval. In: MM, pp. 175–184 (2009)
14. Yi, Y., Zhuang, Y., Wu, F., Pan, Y.: Harmonizing hierarchical manifolds for multimedia document semantics understanding and cross-media retrieval. IEEE Transactions on Multimedia 10(3), 437–446 (2008)

Are Tensor Decomposition Solutions Unique? On the Global Convergence HOSVD and ParaFac Algorithms

Dijun Luo, Chris Ding, and Heng Huang

Department of Computer Science and Engineering
University of Texas, Arlington, Texas, USA

Abstract. Matrix factorizations and tensor decompositions are now widely used in machine learning and data mining. They decompose input matrix and tensor data into matrix factors by optimizing a least square objective function using iterative updating algorithms, *e.g.* HOSVD (High Order Singular Value Decomposition) and ParaFac (Parallel Factors). One fundamental problem of these algorithms remains unsolved: are the solutions found by these algorithms global optimal? Surprisingly, we provide a positive answer for HSOVD and negative answer for ParaFac by combining theoretical analysis and experimental evidence. Our discoveries of this intrinsic property of HOSVD assure us that in real world applications HOSVD provides repeatable and reliable results.

1 Introduction

Tensor based dimension reduction has recently been extensively studied for data mining, pattern recognition, and machine learning applications. Typically, such approaches seek subspaces such that the information are retained while the discarded subspaces contains noises. Most tensor decomposition methods are unsupervised which enable researchers to apply them in any machine learning applications including unsupervised learning and semi-supervised learning. In such applications, one of the central focuses is the uniqueness of the solution. For example, in *missing value completion* problem, such as social recommendation system [1], tensor decompositions are applied to obtain optimal low rank approximation [2]. Since the missing value problem requires iteratively low rank decomposition, the convergence of each iteration is crucial for the whole solution. Other real world applications also highly rely on the stability of the decomposition approaches, such as bioinformatics[3], social network [4], and even marketing analysis [5].

Perhaps High Order Singular Value Decomposition (HOSVD) [6] [7] and Parallel Factors (ParaFac) are some of the most widely used tensor decompositions. Both of them could be viewed as extensions of SVD of a 2D matrix. HOSVD is used in computer vision by Vasilescu and Terzopoulos [8] while ParaFac is used

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 148–159, 2011.

in computer vision by Shashua and Levine [9]. More recently, Yang *et al.* [10] proposed a two dimensional PCA (2DPCA) Ye *et al.* [11] proposed a method called Generalized Low Rank Approximation of Matrices (GLRAM). Both GLRAM and 2DPCA can be viewed in the same framework in 2DSVD (two-dimensional singular value decomposition) [12], and solved by non-iterative algorithm [13]. Similar approaches are also applied as supervised feature extraction [14,15]. The error bounds of HOSVD have been derived [16] and the equivalence between tensor K-means clustering and HOSVD is also established [17].

Although tensor decompositions are now widely used, many of their properties so far have not been well characterized. For example, the tensor rank problem remains a research issue. Counter examples exist that argue against optimal low-dimension approximations of a tensor.

In this paper, we address the solution uniqueness issues[1]. More precisely, non-unique solutions due the existence of large number of local solutions. This problem arises because the tensor decomposition objective functions are non-convex with respect to all the variables and the constraints of the optimization are also non-convex. Standard algorithms to compute these decompositions are iterative improvement. The non-convexity of the optimization implies that the iterated solutions will converge to different solutions if they start from different initial points.

Note that this fundamental uniqueness issue differs from other representation redundancy issues, such as equivalence transformations (i.e. rotational invariance) that change individual factors (U, V, W) but leaves the reconstructed image untouched. These representation redundancy issues can be avoided if we compare different solutions at the level of reconstructed images, rather in the level of individual factors.

The main findings of our investigation are both surprising and comforting. On all real life datasets we tested (we tested 6 data sets and show results for 3 data set due to space limitation), the HOSVD solutions are unique (i.e., different initial starts always converge to an unique global solution); while the ParaFac solution are almost always not unique. Furthermore, even with substantial randomizations (block scramble, pixel scramble, occlusion) of these real datasets, HOSVD converge to unique solution too.

These new findings assure us that in most applications using HOSVD, the solutions are unique — the results are repeatable and reliable.

We also found that whether a HOSVD solution is unique can be reasonably predicted by inspecting the eigenvalue distributions of the correlation matrices involved. Thus the eigenvalue distributions provide a clue about the solution uniqueness or global convergence. We are looking into a theoretical explanation of this rather robust uniqueness of HOSVD.

[1] For matrix and tensor decompositions, there often exists *equivalent* solutions. For example in SVD of $X \cong UV^T$, if (U^*, V^*) is an optimal solution, (U^*R, V^*R) is an equivalent optimal solution, where R is an arbitrary rotational matrix with appropriate dimension. In practice, this problem is fixed by the computational procedure and not considered here.

2 Tensor Decomposition

2.1 High Order SVD (HOSVD)

Consider 3D tensor: $X = \{X_{ijk}\}_{i=1}^{n_1}{}_{j=1}^{n_2}{}_{k=1}^{n_3}$. The objective of HOSVD is to select subspace $U, V,, W$ and core tensor S such that the L_2 reconstruction error is minimized,

$$\min_{U,V,W,S} J_1 = ||X - U \otimes_1 V \otimes_2 W \otimes_3 S||^2 \tag{1}$$

where $U \in \Re^{n_1 \times m_1}$, $V \in \Re^{n_2 \times m_2}$, $W \in \Re^{n_3 \times m_3}$, $S \in \Re^{m_1 \times m_2 \times m_3}$. Using explicit index,

$$J_1 = \sum_{ijk} \Big(X_{ijk} - \sum_{pqr} U_{ip} V_{jq} W_{kr} S_{pqr} \Big)^2. \tag{2}$$

In HOSVD, W, U, V are required to be orthogonal:

$$U^T U = I,\ V^T V = I,\ W^T W = I.$$

With the orthonormality condition, setting $\partial J_1 / \partial S = 0$, we obtain $S = U^T \otimes_1 V^T \otimes_2 W^T \otimes_3 X$, and $J_1 = \|X\|^2 - \|S\|^2$. Thus HOSVD is equivalent to maximize

$$\max_{U,V,W} \|S\|^2 = \|U^T \otimes_1 V^T \otimes_2 W^T \otimes_3 X\|^2 \tag{3}$$

$$= \mathbf{Tr}\ U^T F U \tag{4}$$

$$= \mathbf{Tr}\ V^T G V \tag{5}$$

$$= \mathbf{Tr}\ W^T H W. \tag{6}$$

where

$$F_{ii'} = \sum_{jj'\ell\ell'} X_{ij\ell} X_{i'j'\ell'} (VV^T)_{jj'} (WW^T)_{\ell\ell'} \tag{7a}$$

$$G_{jj'} = \sum_{ii'\ell\ell'} X_{ij\ell} X_{i'j'\ell'} (UU^T)_{ii'} (WW^T)_{\ell\ell'} \tag{7b}$$

$$H_{\ell\ell'} = \sum_{ii'jj'} X_{ij\ell} X_{i'j'\ell'} (UU^T)_{ii'} (VV^T)_{jj'} \tag{7c}$$

Standard HOSVD algorithm starts with initial guess of of (U, V, W) and solve Eqs(3,4,5) alternatively using eigenvectors of the corresponding matrix. Since F, G, H are semi-positive definite, $||S||^2$ are monotonically increase (non-decrease). Thus the algorithm converges to a *local* optimal solution.

HOSVD is a nonconvex optimization problem: The objective function of Eq.(2) w.r.t. (U, V, W) is nonconvex and the orthonormality constraints of Eq.(2) are nonconvex as well. It is well-known that for nonconvex optimization problems, there are many local optimal solutions: starting from different initial guess of (U, V, W), the converged solutions are different. Therefore theoretically, solutions of HOSVD are not unique.

2.2 ParaFac Decomposition

ParaFac decomposition [18,19] is the simplest and also most widely used decomposition model. It approximates the tensor as

$$X \approx \sum_{r=1}^{R} \mathbf{u}^{(r)} \otimes \mathbf{r}^{(r)} \otimes \mathbf{w}^{(r)}, \text{ or } X_{ijk} \approx \sum_{r=1}^{R} U_{ir} V_{jr} W_{kr} \tag{8}$$

where R is the number of factors and $U = (\mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(R)})$, $V = (\mathbf{v}^{(1)}, \cdots, \mathbf{v}^{(R)})$, $W = (\mathbf{w}^{(1)}, \cdots, \mathbf{w}^{(R)})$. ParaFac minimizes the objective

$$J_{\text{ParaFac}} = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} ||X_{ijk} - \sum_{r=1}^{R} U_{ir} V_{jr} W_{kr}||^2 \tag{9}$$

We enforce the implicit constraints that columns of $U = (u^{(1)}, \cdots, u^{(R)})$ are linearly independent; columns of $V = (v^{(1)}, \cdots, v^{(R)})$ are linearly independent; and columns of $W = (w^{(1)}, \cdots, w^{(R)})$ are linearly independent.

Clearly the ParaFac objective function is nonconvex in (U, V, W). The linearly independent constraints are also nonconvex. Therefore, the ParaFac optimization is a nonconvex optimization.

Many different computational algorithms were developed for computing ParaFac. One type of algorithm uses a sequence of rank-1 approximations [20,21,9]. However, the solution of this heuristic approach differ from (local) optimal solutions.

The standard algorithm is to compute one factor at a time in an alternating fashion. The objective decrease monotonically in each step, and the iteration converges to a (local) optimal solution. However, due to the nonconvexity of ParaFac optimization, the converged solution depends heavily on the initial starting point. For this reason, the ParaFac is often not unique.

3 Unique Solution

In this paper, we investigate the problem of whether the solution of a tensor decomposition is unique. This is an important problem, because if the solutions is not unique, then the results are not repeatable and the image retrieval is not reliable.

For a convex optimization problem, there is only one local optimal solution which is also the global optimal solution. For a non-convex optimization problem, there are many (often infinite) local optimal solutions: converged solutions of the HOSVD/ParaFac iterations depend on the initial starting point.

In this paper, we take the experimental approach. For a tensor decomposition we run many runs with dramatically different starting points. If the solutions of all these runs agree with each other (to computer machine precision), then we consider the decomposition has a unique solution.

In the following, we explain the (1) The dramatically different starting point for (U, V, W). (2) Experiments on three different real life data sets. (3) Eigenvalue distributions which can predict the uniques of the HOSVD.

4 A Natural Starting Point for W: The T1 Decomposition and the PCA Solution

In this section, we describe a natural starting point for W. Consider the T1 decomposition [6]

$$X_{ijk} \approx \sum_{k'=1}^{m_3} C_{ijk'} W_{kk'} \quad \text{or} \quad X_{ij}^{(k)} \approx \sum_{k'=1}^{m_3} C_{ij}^{(k')} W_{kk'}. \tag{10}$$

C, W are obtained as the results of the optimization

$$\min_{C,W} J_{\text{T1}} = \sum_{k=1}^{n_3} ||X^{(k)} - \sum_{k'=1}^{m_3} C^{(r)} W_{kk'}||^2. \tag{11}$$

This decomposition can be reformulated as the following:

$$J_{\text{T1}} = ||X||^2 - \mathbf{Tr}\ (W^T \tilde{H} W),\ \tilde{H}_{kk'} = \mathbf{Tr}\ (X^{(k)}[X^{(k')}]^T) = \sum_{ij} X_{ijk} X_{ijk'} \tag{12}$$

C is given by $C^{(r)} = \sum_{k=1}^{n_3} X^{(k)} W_{kr}$. This solution is also the PCA solution. The reason is the following. Let $A = (a_1, \cdots, a_n)$ be a collection of 1D vectors. The corresponding covariance matrix is AA^T and Gram matrix is $A^T A$. Eigenvectors of $A^T A$ are the principal components. Coming back to the T1 decomposition, $\tilde{H}$ is the Gram matrix if we consider each image $X^{(k)}$ as a 1D vector. Solution for W are principal eigenvectors of $\tilde{H}$, which are the principal components.

5 Initialization

For both HOSVD and ParaFac, we generate 7 different initializations:

(R1) Use the PCA results W as explained in §4. Set V to identity matrix (fill zeros in the rest of the matrix to fit the size of $n_2 \times m_2$). This is our standard initialization.

(R2) Generate 3 full-rank matrixes W and V with uniform random numbers of in $(0, 1)$.

(R3) Randomly generate 3 rank deficient matrices W and V with proper size. For first initialization, we randomly pick a column of W and set the column to zero. The rest of columns are randomly generated as in (R2) and the same for V. For second and third initializations, we randomly pick two or three columns of W and set them to zero, and so on. Typically, we use $m_1 = m_2 = m_3 = 5 \simeq 10$. Thus the rank-deficiency at $m_3 = 5$ is strong.

We use the tensor toolbox [22]. The order of update in the alternating updating algorithm is the following: (1) Given (V, W), solve for U (to solve Problem 4); (2) Given (U, W), solve for V (Problem 5); (3) Given (U, V), solve for W (Problem 6); Go back to (1) and so on.

6 Run Statistics and Validation

For each dataset with each parameter setting, we run 10 indepedent tests. For each test, we run HOSVD iterations to convergence (because of the difficulty of estimating convergence criterion, we run total of T=100 iterations of alternating updating which is usually sufficient to converge).

For each independent test, we have 7 different solutions of (U_i, V_i, W_i) where $i = 1, 2, \cdots, 7$ for the solution starting from the i-th initialization. We use the following difference to verify whether the solutions are unique:

$$d(t) = \frac{1}{6} \sum_{i=2}^{7} \left(\|U_i^t - U_1^t\| + \|V_i^t - V_1^t\| + \|W_i^t - W_1^t\| \right),$$

where we introduce the HOSVD iteration index t, and U_i^t, V_i^t, W_i^t are the solution in t-th iteration.

If an optimization problem has a unique solution, $d(t)$ typically starts with nonzero value and gradually decrease to zero. Indeed, this occurs often in Figure 2 The sooner $d(t)$ decreases to zero, the faster the algorithm converges. For example, in the 7th row of Figure 2, the $m_1 = m_2 = m_3 = 5$ parameter setting, the algorithm converges faster than the $m_1 = m_2 = m_3 = 10$ setting.

In our experiments, we do 10 different tests (each with different random starts). If in all 10 tests $d(t)$ decreases to zero, we say the optimization has a unique solution (we say they are globally convergent).

If an optimization has no unique solution (i.e., it has many local optima), $d(t)$ typically remains nonzero at all times, we say the solution of HOSVD is not unique. In Figure 1, we show the results of HOSVD and ParaFac on a random tensor. One can see that in each of the 10 tests, shown as 10 lines in the figure, none of them ever decrease to zero.

For ParaFac we use the difference of reconstructed tensor to evaluate the uniqueness of the solution: $d'(t) = \frac{1}{6} \sum_{i=2}^{7} \|\widehat{X}_i^t - \widehat{X}_1^t\|$, where $\widehat{X}_i^t$ is the reconstruction tensor in the t-th iteration with the i-th starting point. ParaFac algorithm converge slower than HOSVD algorithm. Thus we run 2000 iterations for each test.

7 Eigenvalue Distributions

In these figures, the eigenvalues of F, G, and H are calculated using Eqs. (7a,7b,7c), but setting all UU^T, VV^T, WW^T as identity matrix. The matrices are centered in all indexes. The eigenvalues are sorted and normalized by the sum of the all the eigenvalues. For all F, G, and H, the first eigenvalue is ignored, since it is equivalent to the average along the corresponding index.

8 Datasets

The first image dataset is WANG [23] which contains 10 categories and 100 images for each category. The original size of the image is either 384×256

or 256×384. We select Buildings, Buses, and Food categories and resize the images into a 100×100 size. We also transform all images into 0-255 level gray images. The selected images form a $100 \times 100 \times 300$ tensor. The second dataset is Caltech 101 [24] which contains 101 categories. About 40 to 800 images per category. Most categories have about 50 images. Collected in September 2003 by Li, Andreetto, and Ranzato. The size of each image is roughly 300×200 pixels. We randomly pickup 200 images, resize and transform them into 100×100 0-255 level gray images to form a $100 \times 100 \times 200$ tensor.

9 Image Randomization

Three types randomization are considered: block scramble, pixel scramble and occlusion. In block scramble, an image is divided into n = 2, 4, 8 blocks; blocks are scrambled to form new images (see Figure 2).

In pixel sample, we randomly pick up $\alpha = 40\%, 60\%, 80\%$ of the pixels in the image, and randomly scramble them to form a new image (see Figure 2).

We also experimented with occlusions with sizes up to half of the images. We found that occlusion consistently produce smaller randomization affects and HOSVD results converge to the unique solution. For this reason and the space limitation, we do not show the results here.

10 Main Results

From results shown in Figure 2, we observe the following:

1. For all tested real-life data, ParaFac solutions are not unique, i.e., the converged solution depends on initial starts. This is consistent with the non-convex optimization as explained in §2.2.
2. For all tested real-life data, HOSVD solutions are unique, although theoretically, this is not guaranteed since the optimization of HOSVD is non-convex as explained in §2.1;
3. For even heavily rescrambled (randomized) real-life data, HOSVD solutions are also unique; This is surprising, given that the HOSVD optimization are non-convex.
4. For very severely rescrambled real-life data and pure randomly generated data, HOSVD solutions are not unique.
5. The HOSVD solution for a given dataset may be unique for some parameter setting but non-unique for some other parameter setting.
6. Whether the HOSVD solution for a given dataset will be unique can largely be predicted by inspecting the eigenvalue distribution of the matrices F, G, H. See next section.

11 Eigenvalue-Base Uniqueness Prediction

We found Empirically that the eigenvalue distribution help to predict whether the HOSVD solution on a dataset with a parameter setting is unique or not.

For example, in AT&T dataset HOSVD converges in all parameter settings except in 8×8 block scramble with $m_1 = m_2 = m_3 = 5$. This is because the ignored 3 eigenmodes have very similar eigenvalues as the first five eigenvalues. It is ambiguous for HOSVD to select which of the 8 significant eigenmodes. Thus HOSVD fails to converge to a unique solution.

But when we increase m_1, m_2, m_3 to 10, all 8 significant eigenmodes can be selected and HOSVD converges to a unique solution. This also happens in the other two datasets (see the forth rows in top part of Figure 2.

For 80% pixel scramble in dataset WANG, when $m_1 = m_2 = m_3 = 5, 10$, HOSVD is ambiguous as to select eigenmodes because there are a large number of them with nearly identical eigenvalues around the cutoff. However, if we reduce the dimensions to $m_1 = m_2 = 2, m_3 = 4$ or $m_1 = m_2 = m_3 = 3$, this ambiguity is gone: HOSVD clearly selects the top 2 or 3 eigenmodes. converges (see the last row of the top panel in Figure 2). This same observation also applies to Caltech 101 dataset at 80% pixel scramble in 101 (see the last row of the top part of Figure 2).

For random tensor shown in Figure 1, the eigenvalues are nearly identical to each other. Thus for both parameter setting ($m_1 = m_2 = m_3 = 5$ and $m_1 = m_2 = m_3 = 10$), HOSVD is ambiguous to selection eigenmodes and thus does not converge.

We have also investigated the solution uniqueness problem of the GLRAM tensor decomposition. The results are very close to HOSVD. We skip it due to space limitation.

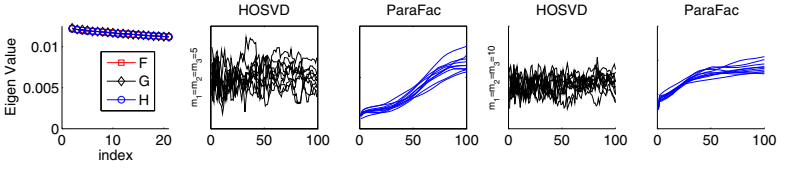


Fig. 1. HOSVD and ParaFace convergence on a $100 \times 100 \times 100$ random tensor

12 Theoretical Analysis

Theoretical analysis of the convergence of HOSVD is difficult due to the fact that U, V, W are orthonormal: they live in Stiefel manifold. Thus the domain of U, V, W are not convex which renders the standard convex analysis not applicable here.

In spite of the difficult, we present two analysis which shed some lights on this global convergence issue.

We consider HOSVD with $m_3 = n_3$ which implies $W = I$. Furthermore, let $X = (X^1 \cdots X^{n_3})$ and we restrict that $X^m \in \Re^{r \times r}$ is symmetric. In this case, V=U, and HOSVD is simplified to

$$\min_U \; J_1 = \sum_m ||X^m - US^m U^T||^2, \; s.t., \; U^T U = I. \tag{13}$$

where $U \in \Re^{r \times k}$ and $S^m \in \Re^{k \times k}$.

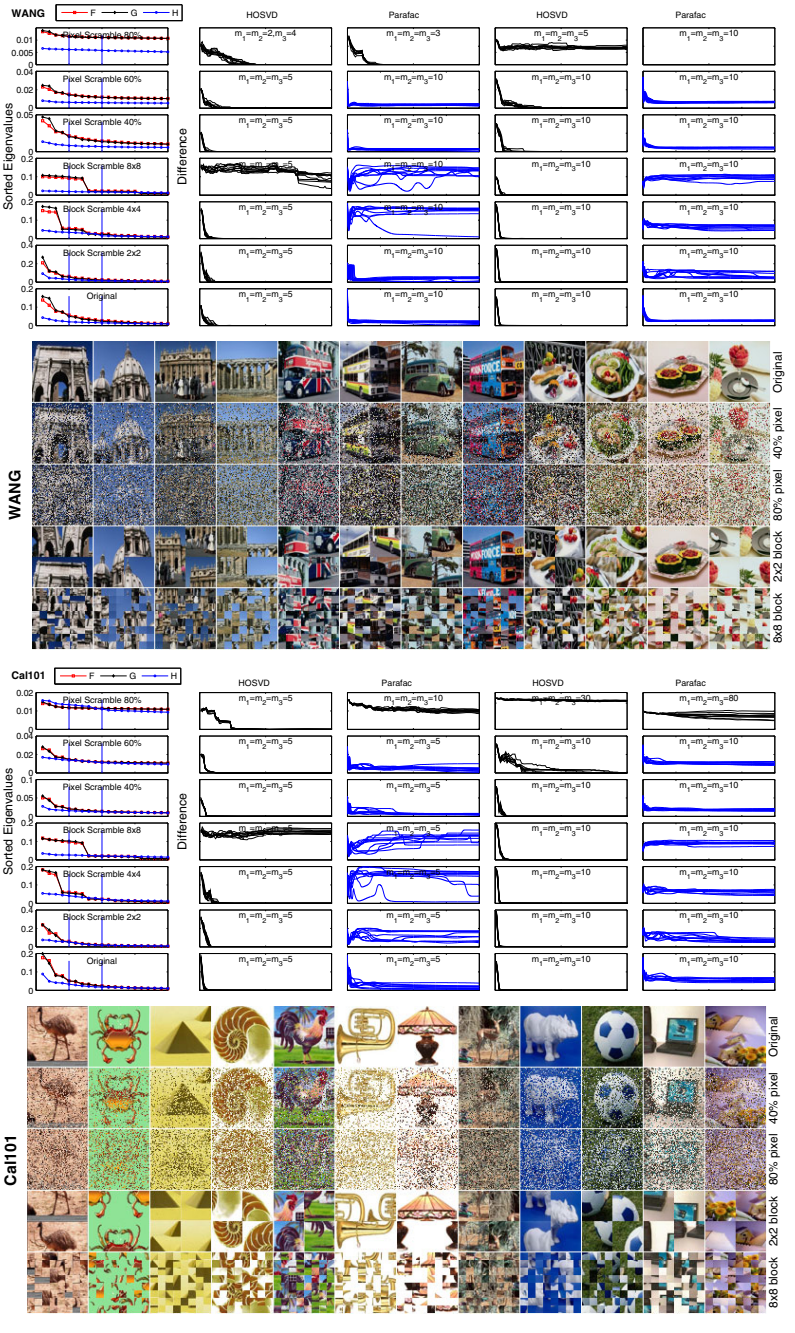


Fig. 2. Convergence analysis for WANG dataset (300 images, 100×100 size for each) and Caltech 101 dataset (200 images of size 100×100 each). Shown are eigenvalues of F, G, H, and solution uniqueness of HOSVD and ParaFac.

At first glance, due to $U^TU = I$. it is hard to prove the global convergence of this problem (convexity). However, we can prove the global convergence using a slightly modified approach.

We can easily show that $S^m = U^T X^m U$, and the optimization becomes

$$\max_U J_2(U) = \sum_m \left(||X^m||^2 - \mathbf{Tr} X^m U U^T X^m U U^T \right)$$

Now let $Z = UU^T$. We study the convexity of

$$\max_Z \; J_2(Z) = \sum_m \mathbf{Tr} X^m Z X^m Z. \tag{14}$$

We now prove

Theorem 1. *The optimization of Eq.(14) is convex when X^m is semi-positive definite(S.P.D.).*

Proof. We have $\frac{\partial J_2}{\partial Z_{ij}} = 2\sum_m \left(X^m Z X^m \right)_{ij}$. The Hessian matrix $H = (H_{[ij][kl]})$ is

$$H_{[ij][kl]} = \frac{\partial^2 J_1}{\partial Z_{ij} \partial Z_{kl}} = 2 \sum_m X^m_{ik} X^m_{lj}.$$

To see if H is s.p.d., we evaluate

$$h \equiv \sum_{ijkl} Z_{ij} H_{[ij][kl]} Z_{kl} = 2 \sum_m \sum_{kl} (Z^T X^m)_{jk} (Z X^m)_{kj}$$

$$= 2 \sum_m \mathbf{Tr}(Z^T X^m Z X^m)$$

Now, every spd matrix X^m can be decomposized into $X^m = B_m^T B_m$. Thus we have

$$h = \mathbf{Tr} Z^T B_m^T B_m Z B_m^T B_m = \mathbf{Tr} B_m Z^T B_m^T B_m Z B_m^T$$

$$= \mathbf{Tr}(B_m Z B_m^T)^T (B_m Z B_m^T) \geq 0$$

Therefore, H is s.p.d. and the optimization of $J_1(U)$ is a convex problem. ■

Indeed, even when X^m are random s.d.p. matrices, the standard HOSVD algorithm convergence to a unique solution no matter what is the starting point.

Next, we consider a nonsymmetric HOSVD problem of Eqs.(4,7) with $F = XVV^TX^T$, i.e., we solve

$$\max_{U,V} \mathbf{Tr}(U^T X V V^T X^T U). \tag{15}$$

We can similary prove

Theorem 2. *The optimization of Eq.(15) is convex.*

Indeed, even when X are random s.d.p. matrices, the standard HOSVD algorithm convergence to a unique solution no matter what is the starting point.

In the simplified HOSVD problems of Eqs.(14,15), we avoided the orthogonality constraints, and thus can prove rigorously the convexity of the optimization problem. In generic HOSVD, the orthogonality constraints cannot be removed and thus the problem is much harder to deal with. We are currently looking into other techniques to analyze the global convergence of HOSVD.

13 Summary

In summary, for all real life datasets we tested, the HOSVD solution are unique (i.e., different initial starts always converge to an unique global solution); while the ParaFac solution are almost always not unique. These finding are new (to the best of our knowledge). They also surprising and comforting. We can be assured that in most applications using HOSVD, the solutions are unique — the results are reliable and repeatable. In the rare cases where the data are highly irregular or severely distored/randomized, our results indicate that we can predict whether HOSVD solution is unique by inspecting the eigenvalue distributions.

Acknowledgement

Dijun Luo was supported by NSF CNS-0923494, NSF IIS-1015219, UTA-REP. Chris Ding and Heng Huang were supported by NSF CCF-0830780, NSF CCF-0939187, NSF CCF-0917274, NSF DMS-0915228.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. 17, 734–749 (2005)
2. Rendle, S., Marinho, L.B., Nanopoulos, A., Schmidt-Thieme, L.: Learning optimal ranking with tensor factorization for tag recommendation. In: KDD, pp. 727–736 (2009)
3. Omberg, L., Golub, G.H., Alter, O.: A tensor higher-order singular value decomposition for integrative analysis of dna microarray data from different studies. Proc. Natl. Acad. Sci. of USA 104, 18371–18376 (2007)
4. Sun, J., Tao, D., Faloutsos, C.: Beyond streams and graphs: Dynamic tensor analysis. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, vol. 12 (2006)
5. Korn, F., Labrinidis, A., Kotidis, Y., Faloutsos, C.: Quantifiable data mining using ratio rules. VLDB Journal: Very Large Data Bases 8, 254–266 (2000)
6. Tucker, L.: Some mathematical notes on three-mode factor analysis. Psychometrika 31, 279–311 (1966)
7. Lathauwer, L.D., Moor, B.D., Vandewalle, J.: A multilinear singular value decomposition. SIAM Journal of Matrix Analysis and Applications 21, 1253–1278 (2000)

8. Vasilescu, M., Terzopoulos, D.: Multilinear analysis of image ensembles: TensorFaces. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 447–460. Springer, Heidelberg (2002)
9. Shashua, A., Levin, A.: Linear image coding for regression and classification using the tensor-rank principle. In: IEEE Conf. on Computer Vision and Pattern Recognition (2001)
10. Yang, J., Zhang, D., Frangi, A.F., Yang, J.: Twodimensional pca: A new approach to appearancebased face representation and recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004)
11. Ye, J.: Generalized low rank approximations of matrices. In: International Conference on Machine Learning (2004)
12. Ding, C., Ye, J.: Two-dimensional singular value decomposition (2dsvd) for 2d maps and images. In: SIAM Int'l Conf. Data Mining, pp. 32–43 (2005)
13. Inoue, K., Urahama, K.: Equivalence of non-iterative algorithms for simultaneous low rank approximations of matrices. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2006)
14. Luo, D., Ding, C., Huang, H.: Symmetric two dimensional linear discriminant analysis (2DLDA). In: CVPR (2009)
15. Nie, F., Xiang, S., Song, Y., Zhang, C.: Extracting the optimal dimensionality for local tensor discriminant analysis. Pattern Recognition 42, 105–114 (2009)
16. Ding, C., Huang, H., Luo, D.: Tensor reduction error analysis – applications to video compression and classification. In: CVPR (2008)
17. Huang, H., Ding, C., Luo, D., Li, T.: Simultaneous tensor subspace selection and clustering: the equivalence of High Order SVD and K-means clustering. In: KDD, pp. 327–335 (2008)
18. Harshman, R.: Foundations of the parafac procedure: Model and conditions for an 'explanatory' multi-mode factor analysis. UCLA Working Papers in Phonetics 16, 1–84 (1970)
19. Carroll, J., Chang, J.: Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition. Psychometrika 35, 283–319 (1970)
20. Zhang, T., Golub, G.H.: Rank-one approximation to high order tensors. SIAM Journal of Matrix Analysis and Applications 23, 534–550 (2001)
21. Kolda, T.: Orthogonal tensor decompositions. SIAM J. Matrix Analysis and App. 23, 243–255 (2001)
22. Bader, B., Kolda, T.: Matlab tensor toolbox version 2.2. (January 2007), http://csmr.ca.sandia.gov/~tgkolda/TensorToolbox/
23. Wang, J.Z., Li, J., Wiederhold, G.: SIMPLIcity: Semantics-sensitive integrated matching for picture LIbraries. IEEE Trans. Pattern Anal. Mach. Intell 23, 947–963 (2001)
24. Perona, P., Fergus, R., Li, F.F.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: Workshop on Generative Model Based Vision, p. 178 (2004)

Improved Spectral Hashing

Sanparith Marukatat and Wasin Sinthupinyo

IMG lab
NECTEC
112 Thailand Science Park,
Klong 1, Klong Luang, Pathumthani, Thailand
{sanparith.marukatat,wasin.sinthupinyo}@nectec.or.th

Abstract. Nearest neighbor search is one of the most fundamental problem in machine learning, machine vision, clustering, information retrieval, etc. To handle a dataset of million or more records, efficient storing and retrieval techniques are needed. Binary code is an efficient method to address these two problems. Recently, the problem of finding good binary code has been formulated and solved, resulting in a technique called spectral hashing [21]. In this work we analyze the spectral hashing, its possible shortcomings and solutions. Experimental results are promising.

1 Introduction

Nearest neighbor search is one of the most fundamental problem in many applications including pattern recognition, image retrieval, object recognition, clustering, etc. For each application, given an object, a set of problem-dependent features is extracted and is used to represent that object. The retrieval process consists in comparing the features vector from the query object and that of the database's objects to find the nearest neighbors or simply near enough neighbors. With today's explosion of digital content the size of the database can be very large. A Naïve comparison cannot be used in practice.

To quickly find nearest neighbors from a very large dataset, there are generally two approaches namely the space partitioning technique and the hashing technique. The first approach aims at organizing the feature space or the dataset, using additional data structure. The simplest organization consists in dividing data into groups or clusters. Then the retrieval process is done in two steps that are the search for the closest cluster and the comparison between the query and each object present in this cluster. Tree structure, e.g. [7,14,5], can be used to speed up the search for closest cluster. However, for dataset which does not have a well-defined cluster structure, one may fail to identify neighbors of query point that falls near the cluster's boundary. To cope with this problem, it is necessary to consider not only the closest cluster but its neighbor clusters as well. As the number of dimension increases, this overhead cost of verifying neighbor clusters increases as well.

To speed-up this search, [2,13] propose to assign a priority to each cluster based on the distance to the query point. This allows accelerating the search but can return only an approximate solution. To further speed-up the search, multiple kd-trees are considered in [19]. Indeed, the authors propose to construct multiple kd-trees with different

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 160–170, 2011.

parameters such that the nodes inside different trees are independent from each other, thus reduce the redundancy in the verification process. Other speed-up method consists in using a more advance tree structure like the cover tree [3] that allow fast rejection of clusters not containing the query point. The lower-bound tree [4] uses information of data in each clusters to estimate the lower bound of the distance between the query point and each point in this cluster in order to quickly discard the clusters candidate while traversing the tree. For ring-cover tree [11], the recursive walk down each node or cluster of the tree depends on the structure of the data in this node, weather it has a clear separable structure (ring node) or it need a subset of elements to cover the whole data (cover node).

Another interesting indexing and search technique called the redundant bits vector or RBV [9] does not use a tree structure for organizing the dataset. Indeed, RBV partitions the feature space into *bins*, each one being identified by a feature and an interval of values. Given an object, a set of corresponding bins can be identified by analyzing its feature's value. For each bin, RBV keeps an *index*, that is a set of objects whose corresponding feature's value falls into the interval of this bin. Given a query object, an intersection between all indexes of the corresponding bins is returned as set of nearest neighbors. A binary string is used to encode index in each bin for fast comparison. Experimentally, we have found that RBV works well if query is indeed in the database or if a nearly duplicated elements exist in the database.

The second approach of fast nearest neighbor search is the hashing technique [12,16,21,20]. The idea is to find a mapping function that transforms each object into a binary code such the Hamming distance between the resulting codes reflects the distance in the original feature space, *e.g.* Euclidean distance. Working with binary code has two obvious advantages. Firstly, it is an efficient method to store million or more data in memory. It is then more scalable for coping with large or huge dataset or even with web-scale dataset. Secondly, the calculation of the distance between two binary codes requires only bit-wise operations and the summation over integer values. These operations are much faster than the floating-point operations. As a consequence the Hamming distance is much cheaper than the other distance. However, finding good binary code that reflects the distance in the original space is a difficult task.

Locality sensitive hashing or LSH [11,8] is one of the most important step in hashing-based technique for fast nearest neighbor search. Indeed, LSH idea is to find a function that yields, with high probability, the same value for neighbor objects and different values for dissimilar objects. In the original paper [11], the authors show that a random bit hashing function can be used to preserve the Hamming distance. Later, it is proven [1], using concept of stable distribution, that one may construct a hashing function *randomly* while preserving the Euclidean distance. Hamming embedding or HE [12] also relies on a random projection to construct a binary code. HE also consider additional information from cluster structure of the dataset while comparing the binary codes.

In practice, the binary code obtained from the random construction, LSH or HE, is not *efficient*, that is large number of hash functions are needed to approximate the Euclidean distance or other distances in the original space. A learning machine can be used to carefully train a compact and efficient binary code as demonstrated in [16]. Followed the same idea, [20] has adopted the boosting similarity sensitive coding technique [18]

and the stacked restricted Boltzmann machines [15] to learn binary codes for a dataset of 12.9 million web images. Even for this very large-scale dataset, the retrieval process using binary code can be done in a fraction of a second [20]. In [21] another approach of find a good binary code has been proposed. Indeed, the authors formally formulate an optimization problem describing the problem of finding a good binary code. The resulting algorithm called *spectral hashing* is simple but produces an efficient code that outperforms prior techniques. This work analyzes the spectral hashing, its possible shortcomings and solutions.

2 Spectral Hashing

2.1 Formulation and Algorithm

Let $\mathbf{x} = [x_1; ...; x_d]^T \in \mathbb{R}^d$ be a feature vectors in d dimensions. Spectral hashing, shorten as SH hereafter, searches for a binary mapping function f minimizing the average distance between the resulting codes with respect to the Euclidean distance in the original space. Formally, this is done by solving the following problem:

$$\min_f \int W(\mathbf{x}, \mathbf{x}')\|f(\mathbf{x}) - f(\mathbf{x}')\|^2 p(\mathbf{x})p(\mathbf{x}')d\mathbf{x}d\mathbf{x}' \tag{1}$$

$$\text{subject to } f(\mathbf{x}) \in \{-1, 1\}^m$$

$$\int f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = 0$$

$$\int f(\mathbf{x})f(\mathbf{x})^T p(\mathbf{x})d\mathbf{x} = I$$

where $W(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/\epsilon^2)$ is the weight function, $p(\mathbf{x})$ is the distribution of $\mathbf{x}$. The second constraint $\int f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = 0$ requires that each bit has equal chance of being 1 or 0, while the third constraint $\int f(\mathbf{x})f(\mathbf{x})^T p(\mathbf{x})d\mathbf{x} = I$ requires that the bits are uncorrelated. By relaxing the first constraint $f(\mathbf{x}) \in \{-1, 1\}^m$, several analytical solutions are possible. These solutions are *eigenfunctions* of the weighted Laplace-Beltrami operator defined on the manifold [21]. More explicitly, let L_p be the weighted Laplacian that maps a function f to $g = L_p f$ with $g(\mathbf{x})/f(\mathbf{x}) = D(\mathbf{x})f(\mathbf{x})p(\mathbf{x}) - \int_{\mathbf{s}} W(\mathbf{s}, \mathbf{x})f(\mathbf{s})p(\mathbf{s})d\mathbf{s}$ and $D(\mathbf{x}) = \int_{\mathbf{s}} W(\mathbf{x}, \mathbf{s})p(\mathbf{s})d\mathbf{s}$. The solutions to the above problem are any functions f satisfying $L_p f = \beta f$ for some real value β, in other word f is an eigenfunction of L_p.

To obtain the solution of the above problem, the original spectral hashing makes two assumptions; 1) $p(\mathbf{x})$ is a separable distribution and 2) each feature is uniformly distributed. The first assumption implies that we may construct an eigenfunction of d-dimensional data by a product of one-dimensional eigenfunctions corresponding to each feature. The second assumption allows us picking the following eigenfunctions as one-dimensional eigenfunctions:

$$\Phi_k(x) = \sin\left(\frac{\pi}{2} + k\pi\frac{(x-a)}{(b-a)}\right) \tag{2}$$

$$\beta_k = 1 - \exp\left(-\frac{\epsilon^2}{2}\left(\frac{k\pi}{b-a}\right)^2\right) \tag{3}$$

where x is a real number uniformly distributed in range $[a, b]$, β_k the corresponding eigenvalues with k a parameter of these eigenfunctions. The multi-dimensional eigenfunction is then constructed implicitly from these one-dimensional eigenfunction.

The above discussion leads to a two steps algorithm; The first step is the principal component analysis (PCA) step and the second one is the eigenfunction selection step. In fact, an example of a separable distribution is a multidimensional Gaussian, once rotated so that the axes are aligned. That is the reason why PCA is used in the first step. Note that for non-vectorial data, kernel PCA [17] can also be used in this step instead. After this step, each object can be represented as a vector in the principal subspace.

The second step consists in evaluating the above eigenvalues (equation (3)) for all possible $k = 1, ..., K$ and for all features $i = 1, ..., l$ with l the dimension of the principal subspace. This result in a list of lK eigenvalues which are sorted in increasing order. Then ignoring the smallest eigenfunction (which corresponds to eigenvalue 0), the next m smallest eigenfunctions are selected to encode object into m bits code. Finally the output from each selected eigenfunction is thresholded to obtain the binary value.

It should be noted that the range of data on each principal axis varies proportional to the variance of the data projected on this axis. It is known that the variance of the i^{th} PCA feature is given by the eigenvalue of this projection axis. As a consequence, the second step of spectral hashing can then be done by sorting eigenfunctions in decreasing order of λ_i/k^2 with $\lambda_1, ..., \lambda_l$ the eigenvalues obtained from the PCA step, then selecting the top m eigenfunctions.

Using the same relation, the eigenfunction (i, k) can be defined on $\mathbb{R}^d$ as follows:

$$\Phi_{i,k}(\mathbf{x}) = \sin\left(\frac{\pi}{2} + k\pi\frac{(\langle \mathbf{v}_i, \mathbf{x}\rangle + 3\sqrt{\lambda_i})}{6\sqrt{\lambda_i}}\right) \tag{4}$$

with $\mathbf{v}_i$ the i^{th} eigenvector, λ_i its corresponding eigenvalue. The binary code is then obtained from $SIGN(\Phi_{i,k}(\mathbf{x}))$.

2.2 Discussion

The spectral hashing relies on the sign of an eigenfunction to encode the projection of data on each principal axis into a binary code. It should be noted that the considered eigenfunction is indeed a sinus function. This function partitions the whole range of data into intervals of equal length and assign the binary code 1 and 0 to these intervals alternately. Uniform distribution implies that the same amount of data falls into each interval. In the following, we shall refer to the amount of data in each interval as its *size*. Thus, uniform distribution assures that the whole range of data is partitioned into intervals not only of the same length but also of the same size. Two solutions are considered in this work. The first one is based on the probability integral transform theorem [6]. The second one relies on manual partition of the data into equal size intervals, then assign the binary code 1 and 0 to these intervals alternately.

Apart from the problem with uniform distribution, the binary encoding with eigenfunction cannot guarantee that the resulting code has equal chance of being 1 or 0. In fact, the parameter k of the eigenfunction partitions the data into $k + 1$ intervals. The first interval is labeled with binary code 1, then the code 0 and 1 are assigned to next intervals

alternately. As a consequence when k is even, there will be more intervals of 1 than that of 0. Thus there will be more 1 than 0 for this bit, even if the data is uniformly distributed.

The above problem can be overcome by first sort the data in increasing order. Then carefully select k thresholds values for partition the whole data into $k+1$ intervals such than the odd (respectively the even) intervals are of the same size and such that the total size of all odd intervals is equal to that of even intervals. While this procedure solves the problem due to the data distribution and equalize the amount of 1 and 0 for the resulting bit, it is inconsistent with the eigenfunction selection procedure. Indeed, the eigenfunction selection criterion is based on the ratio λ_i/k^2 which corresponds roughly to the squared of the length of the interval along the eigen axis $\mathbf{v}_i$ partitioned into $k+1$ equal portions. Having this criterion in mind, the selection of eigenfunction should be done as function of the squared of the average length of the intervals. These different extensions of SH will be studied in next section.

3 Proposed Methods

This section discusses two extensions of SH to handle non-uniform distribution. The first extension (Sect. 3.1) relies on a probabilistic model to transform feature into uniform distribution. The second extension relies manual partition does not require any additional assumption on the data distribution. This technique called *generalized SH*, will be presented in Sect. 3.2.

3.1 SH with Probability Transform

To address the issue of uniform distribution, we propose to further transform the PCA-based feature using the probability integral transform theorem. Indeed, if a random variable Y has continuous accumulated distribution function F, then the probability integral transform theorem states that the new random variable $Z = F(Y)$ will follow uniform distribution in range $[0, 1]$ [6]. Using this fact, one may try to estimate a probabilistic model for each PCA-based feature, then use the corresponding cumulative distribution function (cdf.) to transform this feature into a new feature following uniform distribution.

For instance, let's consider the Gaussian distribution. If y follows Gaussian distribution with mean 0 and variance σ^2 , then its cdf. is given by:

$$\text{cdf}(y;\sigma^2) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{y}{\sqrt{2\sigma^2}}\right)\right]. \tag{5}$$

Using this transformation function $\text{cdf}(y;\sigma^2)$ will be uniformly distributed in $[0, 1]$. For an eigenfunction (i, k), if we assume that the projection of data along the i^{th} principal axis follows Gaussian distribution with 0 means and λ_i variance, then the following eigenfunction can be used instead of $\Phi_{i,k}$ (equation (4)):

$$\Phi^{gauss}_{i,k}(\mathbf{x}) = \sin\left(\frac{\pi}{2} + k\pi\ \text{cdf}(\langle\mathbf{v}_i, \mathbf{x}\rangle; \lambda_i)\right). \tag{6}$$

In the following, we shall refer to the SH using this eigenfunction as the *SH with Gaussian transform*.

3.2 Generalized SH

The figures 1 (a), (b), and (c) show the projection of MNIST data along the first three principal axes. It is clear from this figure that the data distribution is not Gaussian. A better solution would be the use of empirical cdf. similar to the histogram equalization procedure in the image processing [10]. The latter requires, however, a user-selected number of bins for the histogram calculation. A more general solution for partitioning the data into intervals of equal size can be done by first sorting the data in increasing order, then carefully chosen the thresholds based on this sorted list.

This is done by first applying the PCA as in normal SH. Then the projection of data on each principal axis $i = 1, ..., l$ is computed and is sorted in increasing order. Let $z_1^i, ..., z_n^i$ be this sorted list. Then a set of k thresholds $t_1^{i,k}, ..., t_k^{i,k}$ is selected from $z_1^i, ..., z_n^i$ by

$$t_j^{i,k} = z^i_{\frac{jn}{k+1}},\ j = 1, ..., k. \tag{7}$$

The average length of the resulting intervals is then computed as follows:

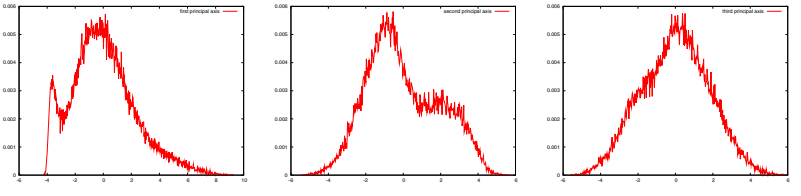
$$len_{i,k} = \frac{1}{(k+1)^2}\left((t_1^{i,k} - z_1^i)^2 + (z_n^i - t_k^{i,k})^2 + \sum_{j=2}^{k}(t_j^{i,k} - t_{j-1}^{i,k})^2\right). \tag{8}$$

The m eigenfunctions having largest average length of intervals $len_{i,k}$ are selected for encoding.

For an eigenfunction (i, k) with $\mathbf{v}_i$ the corresponding eigenvector and $t_1^{i,k}, ..., t_k^{i,k}$ the corresponding thresholds, the output binary code for a new vector $\mathbf{x}$ is computed by

1. Compute z the projection of $\mathbf{x}$ on the axis $\mathbf{v}_i$, i.e. $z = \langle \mathbf{v}_i, \mathbf{x} \rangle$
2. If $z < t_1^{i,k}$ return 1
3. If $t_k^{i,k} \leq z$ return 1 if k is even and 0 if k is odd
4. Otherwise find index j^* such that $t_{j^*-1}^{i,k} \leq z < t_{j^*}^{i,k}$, if j^* is odd then return 1 else return 0.

The final m bits code of the vector $\mathbf{x}$ is the concatenation of the binary code from all m selected eigenfunctions. In the following, this new encoding scheme will be called *generalized SH* or *GSH*.



(a) (b) (c)

Fig. 1. The distribution of the projection of MNIST data along the three first principal axes

3.3 Toward a More Efficient Code

For GSH, for each eigenfunction (i, k) the equality between the number of 1 and 0 can be guarantee by carefully chosen the thresholds $t_j^{i,k},\ j = 1, ..., k$. Indeed, if the data is partitioned into equal size intervals and labeled with binary code 1 and 0 alternately, then when k is even there will be more interval labeled with 1 than that with 0. Therefore, there will be more 1 than 0 for the binary code resulting from this eigenfunction. One may adjust these k thresholds such that

1. the size of all odd intervals are equal to n_o
2. the size of all even intervals are equal to n_e
3. the total size of all odd intervals is equal to the total size of even intervals.

These three requirements can be satisfied by letting

$$n_o = \begin{cases} \frac{n}{k+2} & k \mod 2 = 0 \\ \frac{n}{k+1} & \text{otherwise} \end{cases}$$

$$n_e = \begin{cases} \frac{n}{k} & k \mod 2 = 0 \\ \frac{n}{k+1} & \text{otherwise.} \end{cases}$$

Then the k thresholds are selected by

1. $t^{old} = 0$
2. For $j = 1, ..., k$ do
   (a) $t = t^{old} + \begin{cases} n_e & j \mod 2 = 0 \\ n_o & \text{otherwise} \end{cases}$
   (b) $e_j = z_t^i$
   (c) $t^{old} = t$

with $z_1^i, ..., z_n^i$ be this sorted list of data projected on the axis $\mathbf{v}_i$.

4 Experiments

In this section, we describe the results of several experiments on the nearest neighbor search using binary codes from SH and its extensions.

4.1 Datasets and Evaluation Measures

The evaluation is performed on the MNIST dataset[1]. This is a standard dataset for handwritten digit recognition task. It is composed of 60,000 training examples. Each example, is a bitmap of 28x28 pixels. We simply converted this bitmap into a vector of 784 dimensions. This is a medium size dataset with high dimensional data. Two thousands examples were randomly chosen as test queries.

[1] http://yann.lecun.com/exdb/mnist/

To evaluate different methods, for each test query two sorted lists were computed. The first one was the index of data in the dataset sorted in increasing order of Hamming distance between the databases binary codes and the binary code of the test query. The second one was sorted in increasing order of the Euclidean distance to the test query's features vector. The first 1% in this second sorted list were considered as the true nearest neighbors. By comparing these two lists, a precision/recall graph can be drawn.

4.2 Experimental Results

In this experiment, four methods were compared namely:

1. SH
2. SH with Gaussian transform
3. GSH-1: GSH where all intervals are of the same size
4. GSH-2: GSH where each bit has equal chance of being 1 and 0 (Sect. 3.3).

The first method, the spectral hashing or SH, relies on the uniform data distribution assumption. The second method, SH with Gaussian transform, tries to cope with this problem by using Gaussian assumption. Even if this assumption is also made arbitrary, its parameters are obtained from the actual data. This could lead to an improvement over the normal SH. The next two methods, GSH-1 and GSH-2, try to generalize the idea of SH for any data distribution. GSH-1 relies on the equal-size partition while GSH-2 tries to equalize the number of time each bit is 1 and 0.

Figure 2 shows precision/recall graphs of three methods on MNIST dataset with 8 bits (a), 16 bits (b), 32 bits (c), 64 bits (d), 128 bits (e), and 256 bits (f). From this figure, one may see that the vanilla SH performs surprisingly well compared to other extension, even if this data is clearly not uniformly distributed (see Figure 1). Moreover, one may see that the two extension of SH clearly outperform the normal SH when small number of bits is used (8, 16, or 32 bits). When large number of bits is used (64, 128, and 256 bits), these extensions yield higher precision result while losing the recall.

In fact, different SH extensions tend to produce search result with higher precision but with less recall. This can be seen clearer on the figures 3 (a) and (b). These figures plot precision and recall as function of the thresholds on the Hamming distance to the query using binary code of 32 bits and 256 bits respectively. These results indicate that the precision of the resulting binary code increases as we rectify the underlying assumption about the data distribution. These results also indicate that if we accept all data having Hamming distance to the query object less than a redefined threshold as a neighbor of the query, then the result become more reliable as we move from SH to GSH. For some tasks where precision is more important that recall, e.g. k-NN classifier, GSH represents an interesting technique.

4.3 Computational Cost

Note that using 32-bits code, one will need less than 240 kilobytes to index 60,000 training examples of MNIST dataset. Using this code, on an Intel Core2Duo 1.83Ghz machine the nearest neighbors search is done in roughly 2.3 milliseconds for all methods. This is due to the fact that these methods produces the same type of binary code.

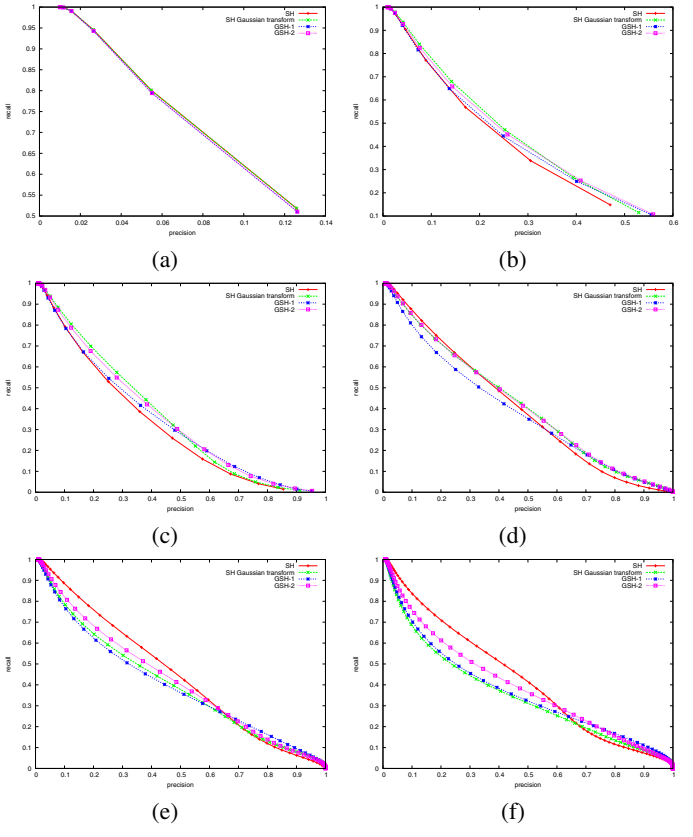


Fig. 2. Precision/recall graphs obtained from the SH and its extension on MNIST dataset with 8 bits (a), 16 bits (b), 32 bits (c), 64 bits (d), 128 bits (e), and 256 bits (f)

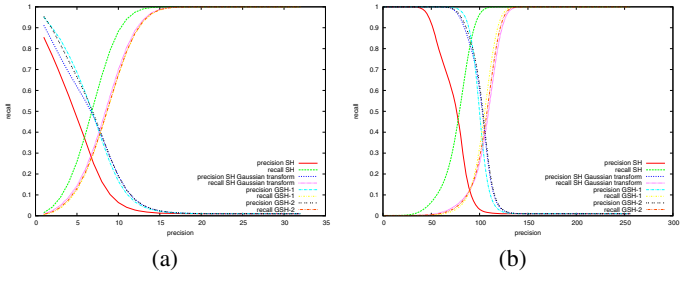


Fig. 3. Plots of precision and recall as function of the thresholds on the Hamming distance. These plots are obtained from the SH and its extensions on MNIST dataset with 32 bits (a), and 256 bits (b).

The difference between these methods lies in the derivation of the eigenfunction. To compute GSH's eigenfunction, the sorted list of data projection on each principal axis is required. As a consequence, it may not be suited for very large dataset. Indeed, we believe that in this case, empirical cdf should be prefered to speed up the calculation.

The retrieval speed can be further improved by *pre-computing* the Hamming distance. In fact, each binary code is kept in the computer's memory as a sequence of *chars*. For example, a 64-bits code can be represented by 8 chars. Each char has 256 possible values. It is then possible to pre-compute the Hamming distance between any two chars and store them in a look-up table. The Hamming distance between two codes is then the summation of the Hamming distance between the chars representing these codes. Using this pre-computed distance and look-up table, the search time for 32-bits code reduces to 1.8 milliseconds.

It is worthy note that the Hamming distance assumes that all bits are of the same importance. In practice, some bit may convey more information that the others; Thus we may assign different weight for each bit in the code. The idea of the pre-computed distance and the look-up table allows computing this weighted Hamming distance without increasing the computational cost. In our preliminary study of this work, we have tried to retrieved nearest neighbors based on this idea with weights selected heuristically. The results were interesting. We are currently working on an automatic selection of these weights.

5 Conclusion and Future Works

In this work, we have analyzed the spectral hashing technique that has been designed to efficiently encode object into a binary string for fast nearest neighbor search. Two extensions of the spectral hashing have been considered. The first one focuses on the uniform distribution assumption. We have shown that by assuming that the data on each principal axis is Gaussian distributed and by using the Gaussian cdf. to further transform this feature, a more efficient code could be obtained. A more general solution called generalized SH is also proposed. These extensions yield search result with more precision than that obtained from the normal SH. Evaluation of these different techniques on larger dataset is currently under investigation.

References

1. Andoni, A., Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.: Locality-Sensitive Hashing Scheme Based on p-Stable Distributions. In: Nearest Neighbor Methods in Learning and Vision: Theory and Practice. MIT Press, Cambridge (2006)
2. Beis, J.S., Lowe, D.G.: Shape indexing using approximating nearest-neighbor search in high-dimensional spaces. In: The International Conference on Computer Vision (CVPR) (1997)
3. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: Proceedings of the International Conference on Machine Learning (ICML) (2006)
4. Chen, Y.-S., Hung, Y.-P., Fuh, C.-S.: Fast algorithm for nearest neighbor search based on a lower bound tree. In: ICCV, pp. 446–453 (2001)
5. Dasgupta, S., Freund, Y.: Random projection trees and low dimensional manifolds. In: Annual ACM Symposium on Theory of Computing (STOC), pp. 537–546 (2008)

6. Devroye, L.: Non-Uniform Random Variate Generation. Springer-Verlag New York Inc., New York (1986)
7. Freidman, J., Bentley, J., Finkel, A.: An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software 3(3), 209–226 (1977)
8. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: VLDB 1999, Proceedings of 25th International Conference on Very Large Data Bases (1999)
9. Goldstein, J., Plat, J.C., Burges, C.J.C.: Redundant bit vectors for quickly searching high-dimensional regions. In: Winkler, J.R., Niranjan, M., Lawrence, N.D. (eds.) Deterministic and Statistical Methods in Machine Learning. LNCS (LNAI), vol. 3635, pp. 137–158. Springer, Heidelberg (2005)
10. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Prentice-Hall, N.J. (2002)
11. Indyk, P., Motwani, R.: Approximate nearest neighbors: Towards removing the curse of dimensionality. In: STOC, pp. 604–613 (1998)
12. Jegou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 304–317. Springer, Heidelberg (2008)
13. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal on Computer Vision (IJCV) 60(2), 91–110 (2004)
14. McNames, J.: A fast nearest-neighbor algorithm based on a principal axis search tree. IEEE Transactions Pattern Analysis and Machine Intelligence 23(9), 964–976 (2001)
15. Salakhutdinov, R.R., Hinton, G.E.: Learning a nonlinear embedding by preserving class neighborhood structure. In: AI and Statistics (2007)
16. Salakhutdinov, R.R., Hinton, G.E.: Semantic hashing. In: SIGIR Workshop on Information Retrieval and Applications of Graphical Models (2007)
17. Scholkopf, B., Smola, A., Muller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10, 1299–1319 (1998)
18. Shakhanarovich, G., Viola, P., Darrell, T.: Fast pose estimation with parameter sensitive hashing. In: International Conference on Computer Vision (ICCV) (2003)
19. Silpa-Anan, C., Hartley, R.: Optimised kd-trees for fast image descriptor matching. In: The International Conference on Computer Vision (CVPR) (2008)
20. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large databases for recognition. In: The International Conference on Computer Vision (CVPR) (2008)
21. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: Advance in Neural Information Processings (NIPS) (2008)

High-Order Co-clustering Text Data on Semantics-Based Representation Model

Liping Jing[1], Jiali Yun[1], Jian Yu[1], and Joshua Huang[2]

[1] School of Computer and Information Technology,
Beijing Jiaotong University, Beijing, China
lpjing@bjtu.edu.cn
[2] Shenzhen Institutes of Advanced Technology, CAS, China

Abstract. The language modeling approach is widely used to improve the performance of text mining in recent years because of its solid theoretical foundation and empirical effectiveness. In essence, this approach centers on the issue of estimating an accurate model by choosing appropriate language models as well as smooth techniques. Semantic smoothing, which incorporates semantic and contextual information into the language models, is effective and potentially significant to improve the performance of text mining. In this paper, we proposed a high-order structure to represent text data by incorporating background knowledge, Wikipedia. The proposed structure consists of three types of objects, term, document and concept. Moreover, we firstly combined the high-order co-clustering algorithm with the proposed model to simultaneously cluster documents, terms and concepts. Experimental results on benchmark data sets (20Newsgroups and Reuters-21578) have shown that our proposed high-order co-clustering on high-order structure outperforms the general co-clustering algorithm on bipartite text data, such as document-term, document-concept and document-(term+concept).

Keywords: Text mining, High-order co-clustering, Representation Model, Semantics, Wikipedia.

1 Introduction

Clustering is an indispensable text mining technique such as query search results organization, automatic abstracting, and etc.. The goal of text clustering is to group documents with similar themes together while separate those with different topics. In order to achieve this goal, the fundamental problem is creating a reasonable and information-abundant representation model. Traditional document representation model in text clustering is term-based vector space model (term VSM) [1]. However, term VSM only covers the syntactic information of a document discarding the semantic information because it assumes all terms independent.

In order to consider the semantic information, latent topic models (e.g., LDA [3] and pLSI [10]) were recently proposed to identify the topics, and terms belonging to one topic are taken to be relevant to each other. This kind of models,

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 171–182, 2011.

to some extent, makes up for the shortage of term VSM, but cannot discover as much semantic information as described in text data only by analyzing syntactic information via statistic methods. Another way to overcome the weakness of the term VSM is incorporating background knowledge (ontology or encyclopedia) into text representation model [11,2,12,19,14,13,15] as concept features to build VSM model.

Even though the background knowledge was used in the literatures [11,12,19,14,13,15] to improve document clustering performance, their final results only contain the document clusters rather than word clusters and even concept clusters. Actually, clustering words is also very important in text clustering besides dividing documents into different groups. Such clustering can be called co-clustering or bi-clustering. Actually, bi-clustering was very popular in biological data analysis [4]. For text mining, Dhillon et al. [6] gave an information theoretic co-clustering to group documents and words simultaneously. However, this simple co-clustering can not deal with complicated data structure, say, heterogeneous data. Gao et al. [8] designed an extended co-clustering with consistent information theory, which can handle heterogeneous data, e.g., documents, terms and document labels. Recently, Dai et al. [5] and Wang et al. [20] applied this idea to self taught learning to do document classification with the aid of auxiliary data.

In this paper, we make use of the background knowledge (Wikipedia) to represent text corpus via a high-order structure. The structure contains three parts, terms, documents and Wikipedia concepts. The Wikipedia concepts will be used to represent a document if they are related to the terms appearing in the given document. Meanwhile, the high-order structure is combined with the high-order co-clustering algorithm [8,9,20] to simultaneously obtain the term groups, concept groups and document groups (We named this kind of co-clustering procedure as HOCOClu). Experimental on real text data have shown that HOCOClu can get better document clustering performance than co-clustering on document-term, document-concept and even document-(term+concept) respectively. Furthermore, the term clusters and concept clusters make sense by investigating the text data.

The rest of the paper is organized as follows: Section 2 introduces the proposed high-order structure by utilizing Wikipedia concepts. In Section 3, the high-order co-clustering (HOCOClu) is firstly combined with the high-order structure to simultaneously find the term group, concept group and document group. The experimental results will be listed and discussed in Section 4. Finally, we conclude the paper in Section 5.

2 High-Order Representation Structure

In this section, we will present a high-order structure to represent document set, as shown in Fig.1. The structure consists of three types of objects: $C = \{c_1, c_2, \cdots, c_p\}$ with p concepts, $D = d_1, d_2, \cdots, d_N$ with N documents, and $T = t_1, t_2, \cdots, t_m$ with m terms. In other words, the documents are represented

with both terms and concepts. In this structure, the relationship between document and term is measured by the term frequency-inverse document frequency ($tf \cdot idf$).

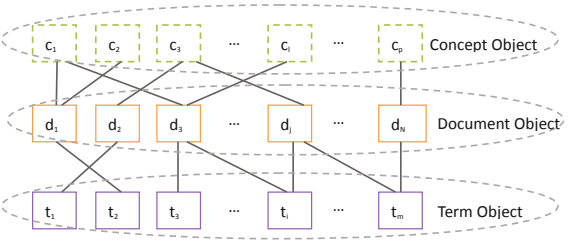


Fig. 1. High-Order Representation Structure

The key step to build the high-order structure is to select the concepts for each document and calculate the contribution of concepts to the document. Wikipedia is, here, used as the concept resources. Given a document, if its term is an anchor in the corresponding Wikipedia articles (in Wikipedia, the title of each article is a concept) [16], this Wikipedia concept is taken to be related to this term. Usually, there are almost twenty Wikipedia concepts related to one term [13]. In this paper, following [17], the most obvious sense (concept) for each term is kept, while the others are removed in order to improve the processing efficiency by discarding insignificant concepts.

Fig.2 gives the framework on how to build the relationship between Wikipedia concept and document. According to the obvious measure, one term can be connected with one Wikipedia concept or not if the term is not an anchor, while one concept can be semantically related to more than one term as shown in Fig.2. Once obtaining the related concepts for each document, we adopt a context-based method [17] to evaluate the semantic relatedness between term and its relevant Wikipedia concept in one document, i.e. R_{ij} in Fig.2.

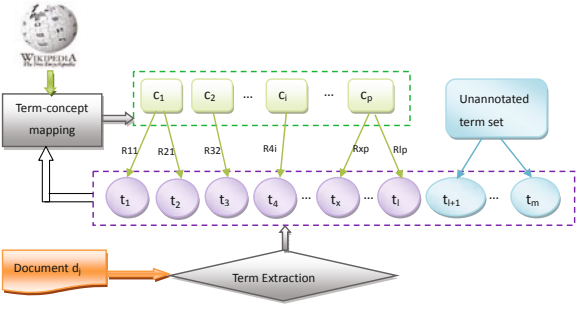


Fig. 2. Mapping from Document terms to Wikipedia concepts

$$Rel(t, c_i|d_j) = \frac{1}{|T|-1} \sum_{t_l \in T \& t_l \neq t} SIM(c_i, c_l) \tag{1}$$

and

$$SIM(c_i, c_l) = 1 - \frac{log(max(|A|, |B|)) - log(|A \cap B|)}{log(|W|) - log(min(|A|, |B|))} \tag{2}$$

where T is the term set of the jth document d_j, t_l is a term in d_j except for t, i.e., $t_l \in T$ and $t_l \neq t$. c_l is the Wikipedia concept related to term t_l. $SIM(c_i, c_l)$ indicates the semantic relatedness between two concepts which is computed according to the hyperlink structure of Wikipedia [18]. In Eq.(2), A and B are the sets of all articles that link to concepts c_i and c_l respectively, and W is the set of all articles in Wikipedia. If A and B do not have any common article, i.e., $|A \cap B| = 0$, we set $SIM(c_i, c_k) = 0$. Eq.(2) is based on term occurrences on Wikipedia-pages. Pages that contain both terms indicate relatedness, while pages with only one of the terms suggest the opposite.

$Rel(t, c_i|d_j)$ indicates the semantic relatedness between term t and concept c_i in document d_j according to term t's context. Higher value of $Rel(t, c_i|d_j)$ shows that concept c_i is more semantically related to term t because c_i is much more similar to the relevant concepts of other terms in d_j (other terms are the context of term t in d_j). Then, the importance of concept c_i in document d_j can be calculated as follows.

$$SAL(c_i, d_j) = \sum_{t \in d_j \& t \sim c_i} w(t, d_j) \times Rel(t, c_i|d_j). \tag{3}$$

where $w(t, d_j)$ is the weight ($tfidf$ is used here) of term t in document d_j. That is, the importance of concept c_i in document d_j is the weighted sum of the related terms' weight based on the semantic relatedness between c_i and its related terms.

So far, a document is represented with a high-order structure which efficiently integrates the syntactic information and semantic information (provided by Wikipedia) and contains three types of objects, term, document and concept. Next, we will make use of this high-order representation structure to simultaneously find the clusters of documents, terms and concepts.

3 High-Order Co-clustering Method

In this section, we will show how to determine tripartite clusters based on the high-order representation structure. Gao et al. [8,9] proposed a kind of co-clustering methods to cluster high-order heterogeneous data. Among them, CIT [9] and CoCC[5,20] were based on consistency information theory [6] and proven to be more efficient and effective than CBGC [8]. All these three methods are used on the heterogeneous text data with category-document-term format, here category is the document category label. However, in real applications, very few documents have category information. In this section, we will show how to apply the consistency information theory on text data represented via high-order structure to identify document clusters, term clusters and concept clusters simultaneously. Also, we adopted a more effective optimizing process [5] to implement

this high-order co-clustering, named as HOCOClu. To the best of our knowledge, our work is the first one to use high-order co-clustering to find concepts, documents and terms clusters simultaneously.

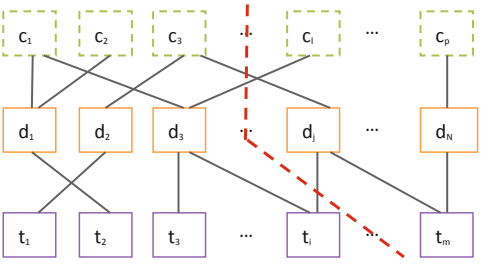


Fig. 3. Tripartite graph of concept-document-term

The goal of HOCOClu can be taken as finding the tripartite graph of the high-order representation, as shown in Fig.3. With this tripartite graph, we can group objects C, D and T into different clusters denoted as $\hat{C}$, $\hat{D}$, and $\hat{T}$ respectively, e.g., the data objects are separated by the dotted line. The whole clustering procedure can be implemented via the consistency information theory [6] by optimizing the following objective function.

$$\begin{aligned} F(C,D,T) &= (I(C,D) - I(\hat{C},\hat{D})) + \lambda(I(T,D) - I(\hat{T},\hat{D})) \\ &= D_{KL}(p_1(C,D)||q_1(C,D)) + \lambda D_{KL}(p_2(T,D)||q_2(T,D)) \end{aligned} \tag{4}$$

where λ is a trade-off parameter to balance the influence between the syntactic information and the semantic information. $D_{KL}(p||q)$ is the KL-divergence. $p(d_j, c_l)$ indicates the joint probability between concept c_l and document d_j and is calculated via $p(d_j, c_l) = \frac{v(d_j,c_l)}{\sum_{i=1}^{p}\sum_{j=1}^{N} v(d_j,c_l)}$. $v(d_j, c_l)$, computed with Eq.(3), is the weight of concept c_l in document d_j. Similarly, $p(d_j, t_i)$ indicates the joint probability between term t_i and document d_j, where $v(d_j, t_i)$ is $tf \cdot idf$ value when calculating $p(d_j, t_i)$.

In Eq.(4), $q_1(C, D)$ is the distribution between concept cluster and document cluster, similarly, $q_2(T, D)$ for term cluster and document cluster, which are formulated as follows.

$$q_1(C,D) = p_1(\hat{C},\hat{D})p_1(C|\hat{C})p_1(D|\hat{D}) \quad C \in \hat{C} \quad D \in \hat{D} \tag{5}$$

and

$$q_2(T,D) = p_2(\hat{T},\hat{D})p_2(T|\hat{T})p_2(D|\hat{D}) \quad T \in \hat{T} \quad D \in \hat{D} \tag{6}$$

where $\hat{T}$, $\hat{D}$, and $\hat{C}$ are the group of terms, document, and concept which are iteratively choose via

$$C_C(c) = arg\min_{\hat{c}\in\hat{C}} D(p_1(D|c)||q_1(D|\hat{c})) \tag{7}$$

$$C_T(t) = arg\min_{\hat{t}\in\hat{T}} D(p_2(D|t)||q_2(D|\hat{t})) \tag{8}$$

and

$$C_D(d) = arg\min_{\hat{d}\in\hat{D}}(p_1(d)D(p_1(C|d)||q_1(C|\hat{d})) + \lambda p_2(d)D(p_2(T|d)||q_2(T|\hat{d}))) \quad (9)$$

Eq.(7-9) are conditions to appropriately cluster concepts, terms and documents respectively. Therefore, Eq.(4) can be taken as a consistent fusion of two pairwise co-clustering sub-problems (D-C co-clustering and D-T co-clustering) by sharing the same document variable D and document clustering result $\hat{D}$. Such consistent fusion can make the overall partitioning be globally optimal under objective function Eq.(4). Based on the above definition, the best $\hat{T}$, $\hat{D}$, and $\hat{C}$ can be simultaneously returned by Algorithm 1.

Algorithm 1. HOCOClu Algorithm

input : Text data set probability distribution ($p_1(C, D)$, $p_2(T, D)$) under high-order structure; Parameter λ; Number of iterations H; Number of Term clusters k_T, number of concept clusters k_C and number of document clusters k_D

output: The final clustering result $\hat{T}$, $\hat{D}$, and $\hat{C}$

{*Initialize cluster distribution*}

Initialize term clusters, concept clusters and document clusters, i.e., set $C_T^{(0)}(t)$, $C_C^{(0)}(c)$, and $C_D^{(0)}(d)$.

{*Compute the initial posterior probability distribution*}

Compute $q_1^{(0)}(C, D)$ and $q_2^{(0)}(T, D)$ based on initialization $\hat{T}$, $\hat{C}$ and $\hat{D}$ with Eq.(5) and Eq.(6) respectively.

{*Iteratively update* $C_C(c)$, $C_T(t)$ *and* $C_D(d)$}

for $\forall h = 1, \cdots, H$ **do**

  Update $C_C^{(h)}(c)$ based on $p_1(C, D)$, $q_1^{(t-1)}(C, D)$ and Eq.(7).

  Update $C_T^{(h)}(t)$ based on $p_2(T, D)$, $q_2^{(t-1)}(T, D)$ and Eq.(8).

  Update $C_D^{(h)}(d)$ based on $p_1(C, D)$, $p_2(T, D)$, $q_1^{(h-1)}(C, D)$, $q_2^{(h-1)}(T, D)$ and Eq.(9).

  Update $q_1^{(h)}(C, D)$ based on $p_1(C, D)$, $C_C^{(h)}(c)$, $C_D^{(h)}(d)$ and Eq.(5).

  Update $q_2^{(h)}(T, D)$ based on $p_2(C, D)$, $C_T^{(h)}(t)$, $C_D^{(h)}(d)$ and Eq.(6).

end

Return $C_C^{(H)}$, $C_T^{(H)}$ and $C_D^{(H)}$ as the final clustering results on the concepts, terms and documents respectively.

4 Experimental Results and Discussion

4.1 Dataset

In this section, we would test our proposed highi-order representation structure and HOCOClu co-clustering on real data sets (20Newsgroups and Reuters-21578) with the aid of Wikipedia. Five test sets were created from

20Newsgroups. 20NG-Diff4 with four substatially different classes, and 20NG-Sim4 with four similar classes with 1000 documents in each class. Another three data sets, Binary, Multi5 and Multi10 were created following [6]. Among them, Binary contains two similar classes with 250 documents in each class, Multi5 covers five categories with 100 documents per category, and Multi10 includes ten groups with 50 documents per group. Reuters-21578 consists of short news articles dating back to 1987. We created the R_Min20Max200 set following [14] which contains the Reuters categories that have at lest 20 and at most 200 documents and R_Top 10 consisting of the largest 10 classes. All these seven data sets in Table 1 only contain the single-label documents. For the multi-label document clustering, we will study in the future.

Table 1. Data set summary

Dataset	Description	♯Classes	♯Documents	♯Words	♯Concepts	♯Words no Concepts
D1	Binary	2	500	2000	1649	204
D2	Multi5	5	500	2000	1667	209
D3	Multi10	10	500	2000	1658	194
D4	20NG-Diff4	4	4000	5433	4362	503
D5	20NG-Sim4	4	4000	4352	3502	426
D6	R_Min20Max200	25	1413	2904	2450	176
D7	R_Top 10	10	8023	5146	4109	448

For each data set, the words were extracted by preprocessing steps, selecting only alphabetical sequences, stemming them and removing stop words. For data sets 20NG-Diff4, 20NG-Sim4, R_Min20Max200 and R_Top 10, we removed the infrequent words that appeared in less three documents. For data sets Binary, Multi5 and Multi10, we adopted the feature selection method following [6], i.e., selecting the top 2000 words according to the MI score between words and corpus, so that it is more reasonable to compare with the clustering result of the original co-clustering algorithm proposed in [6]. The Wikipedia concepts for these words in each data set were determined via the method in Section 2.

4.2 Clustering Results and Discussion

Since the category of each document was known in these data sets, we used the external cluster validation method to evaluate the clustering results by calculating the correspondence between the clusters generated by a clustering algorithm and the inherent categories of the data set. Table 2 lists the three evaluation functions, *Entropy* [21], *FScore* [21] and the normalized mutual information (*NMI*) [22], which were used to evaluate clustering results.These functions can be interpreted as follows. The smaller the *Entropy*, the better the clustering performance. The larger the *FScore*/*NMI*, the better the clustering performance.

Table 2. Evaluation Functions

Entropy	$\sum_{l=1}^{k} \frac{n_l}{n} \left(-\frac{1}{\log k} \sum_{j=1}^{k} \frac{n_{jl}}{n_l} \cdot \log \frac{n_{jl}}{n_l} \right)$
FScore	$\sum_{j=1}^{k} \frac{n_j}{n} \cdot \max_{1 \le l \le k} \left\{ \frac{2 \cdot n_{jl}/n_h \cdot n_{jl}/n_l}{n_{jl}/n_j + n_{jl}/n_l} \right\}$
NMI	$\frac{\sum_{j,l} n_{jl} log\left(\frac{n \cdot n_{jl}}{n_j n_l}\right)}{\sqrt{(\sum_j n_j log \frac{n_j}{n})(\sum_l n_l log \frac{n_l}{n})}}$

n_j, n_l are the numbers of documents in class L_j and in cluster C_l respectively
n_{jl} is the number of documents occurring in both class L_j and cluster C_l
n is the total number of documents in the data set, and k is the number of classes.

Next, we will demonstrate the performance of HOCOClu on text data represented via high-order structure by comparing with the original co-clustering on document-term (D-T), document-concept (D-C) and document-(term+concept) (D-(T+C)) models.

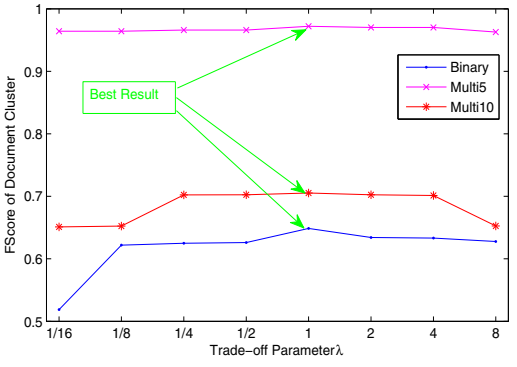


Fig. 4. The document clustering FScore curves as a function of trade-off parameter λ

In the experiments, the term clusters and concept clusters were initialized by k-means clustering algorithm on term-document TFIDF matrix and concept-document TFIDF matrix respectively. For document clusters initialization, in co-clustering of D-T, document clusters were obtained by k-means on document-term TFIDF matrix, while, in co-clustering of D-C, document clusters were got by k-means on document-concept TFIDF marix. In HOCOClu, we ensembled the document clustering results of k-means on the above two matrices with the method proposed in [7]. The number of document clusters is set to be the true number of classes. For term and concept, we set the number of term clusters and concept clusters with different values from $m/10$ to $m/100$ (m is the number of terms or concepts), finally, the best clustering result for each dataset was shown.

For trade-off parameter λ in Eq.(4), we constructed a series of experiments to show how λ affects the clustering performance. Fig.4 presents the document clustering *FScore* curve given by HOCOClu along with changing λ on three small data sets. From this figure, it can be seen that, when λ decreases, implying the weights of the concept (i.e., semantic) information being lower, the

performance of HOCOClu declines. On the other hand, when λ is sufficiently large, i.e., $\lambda > 1$, meaning the weights of the term (i.e., syntactic) information being lower, the performance of HOCOClu also decreases slightly. This indicates that clustering performance benefits from both term and concept information, esp., when treating syntactic information and semantic information equally. In the following experiments, we set the trade-off parameter λ to be 1, which is the best point in Fig.4.

Table 3. Comparison of the co-clustering (COClu) performance with different strategies and HOCOClu clustering on High-order representation structure (Bold-face numbers indicate the best evaluation result in the four models)

		Dataset	D1	D2	D3	D4	D5	D6	D7
COClu	D-T	EN	0.8513	0.3457	0.5397	0.1708	0.5546	0.2764	0.2180
		F	0.6342	0.7717	0.5804	0.8967	0.6851	0.6242	0.5931
		NMI	0.1761	0.6647	0.4841	0.8507	0.5141	0.6928	0.5481
	D-C	EN	0.8474	0.3544	0.4910	0.1740	0.5592	0.2712	0.2221
		F	0.6332	0.7526	0.6109	0.8948	0.6722	0.6221	0.5764
		NMI	**0.1817**	0.6595	0.5402	0.8471	0.5142	0.6989	0.5408
	D-(T+C)	EN	**0.8586**	0.3501	0.5259	0.1651	0.5547	0.2749	0.2219
		F	0.6261	0.7713	0.6015	0.8980	0.6765	0.6091	0.5642
		NMI	0.1689	0.6626	0.5068	0.8564	0.5179	0.6956	0.5435
HOCOClu		EN	0.8548	**0.0833**	**0.4017**	**0.1547**	**0.4811**	**0.1936**	**0.1975**
		F	**0.6487**	**0.9721**	**0.7053**	**0.9016**	**0.7359**	**0.6927**	**0.6019**
		NMI	0.1723	**0.9170**	**0.6039**	**0.8676**	**0.5851**	**0.7797**	**0.5699**

Table 3 gives the comparison clustering results on seven data sets. Three figures in each cell represent the values of *Entropy* (En), *FScore* (F) and *NMI* respectively. The good clustering results are marked in the bold case. We can see that the proposed HOCOClu on high-order structure can achieve the better clustering performance than all co-clustering results on D-T, D-C and D-(T+C), because HOCOClu efficiently makes use of the term and concept information than the other three methods.

Meanwhile, HOCOClu has ability to identify the word and concept clusters. Next, we will take Multi5 data set as an example to show the advantage of HOCOClu.

Table 4. Confusion matrix obtained by HOCOClu on Multi5

Category Description	$C1_D$	$C2_D$	$C3_D$	$C4_D$	$C5_D$
comp.graphics	0	0	4	96	0
rec.motorcycles	0	0	99	1	0
rec.sport.baseball	0	95	4	1	0
sci.space	95	0	1	2	2
talk.politics.mideast	0	0	0	0	100

Table 5. Term and concept clusters IDs corresponding to Multi5 document clusters identified by HOCOClu

DocClu ID	Word Cluster (WC) IDs	Concept Cluster (CC) IDs
$C1_D$	2,6,8,10,12,13,17,18,19, 20,21,24,32,37,39	2,4,7,9,11,15,19,20,21, 23,25,27,30,34,35,38
$C2_D$	11,16,25,34,35,42,44	13,24,31,37,39,47
$C3_D$	7,22,28,31,36,46,48,49	5,22,43,48,49,50
$C4_D$	1,23,29,30,33,38,45,47	1,26,28,32,33,44,45,46
$C5_D$	3,4,5,9,14,15,26,27,40, 41,43,50	3,6,8,10,12,14,16,17,18, 29,36,40,41,42

Table 4 indicates the confusion matrix of Multi5 obtained by HOCOClu. In our experiments, the document clustering performance was best when the number of word clusters and number of concept clusters were set to be 50 (i.e., 2000/40). From Table 4, we can see document cluster $C1_D$ talks about *sci.space*, $C2_D$ about *rec.sport.baseball*, $C3_D$ mainly about *rec.motorcycles*, $C4_D$ mainly about *comp.graphics* and $C5_D$ about *talk.politics.mideast*.

Table 5 gives the term cluster IDs and concept cluster IDs corresponding to each Multi5 document cluster. In order to check whether the word cluster or concept cluster is related to the document cluster (i.e., related to the cluster topic), we list ten words and five concepts (separated by space) of each representative word or concept cluster (two word clusters and two concept clusters shown here) for Multi5 in Table 6. Obviously, the words or concepts in the representative clusters are semantically related to the topic of document clusters.

Table 6. Representative term and concept clusters corresponding to Multi5 document clusters identified by HOCOClu

DocClu	Term Clu and Concept Clu	
$C1_D$	WC18	space orbit probe mission launch earth satellite astronaut moon lunar
	WC21	sci comet rocket flight constant wing energy material aircraft global
	CC20	Science Natural-environment Remote-(band) Chemical-element Aircraft
	CC25	Space-Shuttle Mercury-(planet) Lander-(spacecraft) Atmosphere Astronomy
$C2_D$	WC42	game baseball player sport team netcom season pitch blue hitter
	WC44	win catcher rbi score sandberg bat brave cub era rookie
	CC24	Sports-league Closer-(baseball) Netcom-(USA) pitch-(baseball) Team
	CC37	Midway-(fair) Pennant-(sports) Player-(game) Win-(baseball) Batting-average
$C3_D$	WC31	motorcycle ride bike drink aber denizen nec rider bmw uknet
	WC46	rec mike tech arizona roger glove fan spring phoenix quick
	CC22	Motorcycle Bicycle Rider Alcohol Leather
	CC43	Technology Rogers-Communications Spring Fasting Speed
$C4_D$	WC23	sgi color algorithm visual postscript bounce op symposium siemen monitor
	WC29	compute ibm bit level server user keyword version library band
	CC28	Code Program-Management Window Interface-(computer science) Visible-spectrum
	CC33	Keyword-(computer programming) Manufacturing Library Computer Pixel
$C5_D$	WC3	armenian isra turkish arab israel party mideast jew hasan turk
	WC43	politic talk libertarian peace leader woman border negotiate vote commit
	CC8	Terrorism Karabakh Baku-(spirit) Palestine Religion
	CC36	Impressment Injury Count Agreement-(linguistics) Monarchy

5 Conclusions and Future Work

In this paper, we presented a semantics-based high-order structure for text representation with the aid of Wikipedia. The proposed structure contains three types of objects, term, document and concept. The contribution of concept to document is calculated based on the semantic relatedness between concept and its related terms in the current document. This proposed structure takes into account both semantic information and syntactic information. Combining the high-order co-clustering algorithm, the clusters of terms, documents and concepts can be simultaneously identified. Experimental results on real data set have shown that the high-order co-clustering on high-order representation structure outperforms the co-clustering on document-term, document-concept and document-(term+concept) model.

Even though the concept information was used in the proposed method, there is abundant information in Wikipedia, e.g., category information. Such category information is also helpful in text clustering [13]. In the future, we plan to build text representation model by integrating Wikipedia category, so that the relationship between Wikipedia concepts could be taken into account.

Acknowledgement

This work was supported by the Fundamental Research Funds for the Central Universities (2010RC029), the National Natural Science Foundation of China (60905028, 90820013, 60875031, 61033013), 973 project (2007CB311002) and the Project-sponsored by SRF for ROCS, SEM.

References

1. Yates, R., Neto, B.: Modern information retrieval. Addison-Wesley Longman, Amsterdam (1999)
2. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering short texts using wikipedia. In: Proc. of the 30th ACM SIGIR, pp. 787–788 (2007)
3. Blei, D., Ng, A., Jordan, M.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
4. Busygin, S., Prokopyev, O., Pardalos, P.: Bi-clustering in data mining. Computers & Operations Research 35, 2964–2967 (2008)
5. Dai, W., Yang, Q., Xue, G., Yu, Y.: Self-taught clustering. In: Proc. of the 25th ICML (2008)
6. Dhillon, I., Mallela, S., Modha, D.: Information-theoretic co-clustering. In: Proc. of the 9th ACM SIGKDD, pp. 89–98 (2003)
7. Fred, A.: Finding consistent clusters in data partitions. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, pp. 309–318. Springer, Heidelberg (2001)
8. Gao, B., Liu, T., Cheng, Q., Ma, W.: Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In: Proc. of ACM SIGKDD, pp. 41–50 (2005)
9. Gao, B., Liu, T., Ma, W.: Star-structured high-order heterogeneous data co-clustering based on consistent information theory. In: Proc. of the 6th ICDM (2006)

10. Hofmann, T.: Probabilistic latent semantic indexing. In: Proc. of ACM SIGIR, pp. 50–57 (1999)
11. Hotho, A., Staab, S., Stumme, G.: Wordnet improves text document clustering. In: Proc. of the Semantic Web Workshop at the 26th ACM SIGIR, pp. 541–544 (2003)
12. Hu, J., Fang, L., Cao, Y., Zeng, H., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging wikipedia semantics. In: Proc. of the 31st ACM SIGIR, pp. 179–186 (2008)
13. Hu, X., Zhang, X., Lu, C., Park, E., Zhou, X.: Exploiting wikipedia as external knowledge for document clustering. In: Proc. of the 15th ACM SIGKDD, pp. 389–396 (2009)
14. Huang, A., Milne, D., Frank, E., Witten, I.: Clustering documents using a wikipedia-based concept representation. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 628–636. Springer, Heidelberg (2009)
15. Jing, L., Ng, M., Huang, J.: Knowledge-based vector space model for text clustering. Knowledge and Information Systems 25, 35–55 (2010)
16. Kittur, A., Chi, E., Suh, B.: What's in wikipedia? mapping topics and conflict using socially annotated category structure. In: Proc. of the 27th CHI, pp. 1509–1512 (2009)
17. Medelyan, O., Witten, I., Milne, D.: Topic indexing with wikipedia. In: Proc. of AAAI (2008)
18. Milne, D., Witten, I.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: Proc. of the Workshop on Wikipedia and Artificial Intelligence at AAAI, pp. 25–30 (2008)
19. Wang, P., Domeniconi, C.: Building semantic kernels for text classification using wikipedia. In: Proc. of the 14th ACM SIGKDD, New York, NY, USA, pp. 713–721 (2008)
20. Wang, P., Domeniconi, C., Hu, J.: Using wikipedia for co-clustering based cross-domain text classification. In: Proc. of the 8th ICDM, pp. 1085–1090 (2008)
21. Zhao, Y., Karypis, G.: Comparison of agglomerative and partitional document clustering algorithms. Technical Report 02-014, University of Minnesota (2002)
22. Zhong, S., Ghosh, J.: A comparative study of generative models for document clustering. In: Proc. of SDW Workshop on Clustering High Dimensional Data and its Applications, San Francisco, CA (2003)

The Role of Hubness in Clustering High-Dimensional Data

Nenad Tomašev[1], Miloš Radovanović[2], Dunja Mladenić[1], and Mirjana Ivanović[2]

[1] Institute Jožef Stefan
Artificial Intelligence Laboratory
Jamova 39, 1000 Ljubljana, Slovenia
nenad.tomasev@ijs.si, dunja.mladenic@ijs.si
[2] University of Novi Sad
Department of Mathematics and Informatics
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
radacha@dmi.uns.ac.rs, mira@dmi.uns.ac.rs

Abstract. High-dimensional data arise naturally in many domains, and have regularly presented a great challenge for traditional data-mining techniques, both in terms of effectiveness and efficiency. Clustering becomes difficult due to the increasing sparsity of such data, as well as the increasing difficulty in distinguishing distances between data points. In this paper we take a novel perspective on the problem of clustering high-dimensional data. Instead of attempting to avoid the curse of dimensionality by observing a lower-dimensional feature subspace, we embrace dimensionality by taking advantage of some inherently high-dimensional phenomena. More specifically, we show that hubness, i.e., the tendency of high-dimensional data to contain points (hubs) that frequently occur in k-nearest neighbor lists of other points, can be successfully exploited in clustering. We validate our hypothesis by proposing several hubness-based clustering algorithms and testing them on high-dimensional data. Experimental results demonstrate good performance of our algorithms in multiple settings, particularly in the presence of large quantities of noise.

1 Introduction

Clustering in general is an unsupervised process of grouping elements together, so that elements assigned to the same cluster are more similar to each other than to the remaining data points [1]. This goal is often difficult to achieve in practice. Over the years, various clustering algorithms have been proposed, which can be roughly divided into four groups: *partitional*, *hierarchical*, *density-based*, and *subspace* algorithms. Algorithms from the fourth group search for clusters in some lower-dimensional projection of the original data, and have been generally preferred when dealing with data that is high-dimensional [2,3,4,5]. The motivation for this preference lies in the observation that having more dimensions usually leads to the so-called *curse of dimensionality*, where the performance of many standard machine-learning algorithms becomes impaired. This is mostly due to two pervasive effects: the empty space phenomenon and concentration of distances. The former refers to the fact that all high-dimensional data

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 183–195, 2011.

sets tend to be sparse, because the number of points required to represent any distribution grows exponentially with the number of dimensions. This leads to bad density estimates for high-dimensional data, causing difficulties for density-based approaches. The latter is a somewhat counterintuitive property of high-dimensional data representations, where all distances between data points tend to become harder to distinguish as dimensionality increases, which can give rise to problems with distance-based algorithms [6,7,8].

The difficulties in dealing with high-dimensional data are omnipresent and abundant. However, not all phenomena which arise are necessarily detrimental to clustering techniques. We will show in this paper that *hubness*, which is the tendency of some data points in high-dimensional data sets to occur much more frequently in k-nearest neighbor lists of other points than the rest of the points from the set, can in fact be used for clustering. To our knowledge, this has not been previously attempted. In a limited sense, hubs in graphs have been used to represent typical word meanings in [9]. This, however, was not used for data clustering. Therefore, we focused first of all on exploring the potential value of using hub points in clustering by constructing hubness-based clustering algorithms and testing them in high-dimensional settings. The hubness phenomenon and its relation to clustering will be further addressed in Section 3.

The rest of the paper is structured as follows. In the next section we present related work, Section 3 discusses in general the phenomenon of hubness, while Section 4 describes the proposed algorithms that are exploiting hubness for data clustering. Section 5 presents the experiments we performed on both synthetic and real world data, and in Section 6 we give our final remarks.

2 Related Work

Even though hubness has not been given much attention in data clustering, hubness information is drawn from k-nearest-neighbor lists, which have been used in the past to perform clustering in various ways. These lists may be used for computing density estimates, by observing the volume of space determined by the k nearest neighbors. Density-based clustering methods often rely on this kind of density estimation [10,11,12]. The implicit assumption made by density-based algorithms is that clusters exist as high-density regions separated from each other by low-density regions. In high-dimensional spaces this is often difficult to estimate, due to data being very sparse. There is also the issue of choosing the proper neighborhood size, since both small and large values of k can cause problems for density-based approaches [13]. Enforcing k-nearest-neighbor consistency in algorithms such as K-means was also experimented with [14]. This approach proposed moving closed neighbor-sets between clusters in iterations instead of using single data points. However, the most typical usage of k-nearest-neighbor lists relates to constructing a k-NN graph, where nodes are connected by an edge if one of them is in the k-nearest-neighbor list of the other [15]. The problem is then reduced to graph clustering, with a number of approaches available.

3 The Hubness Phenomenon

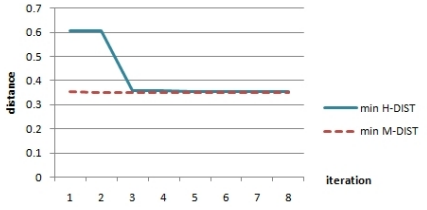
Hubness is an aspect of the curse of dimensionality pertaining to nearest neighbors which has only recently come to attention, unlike the much discussed distance concentration phenomenon. Let $D \subset \mathbb{R}^d$ be a set of data points and let $N_k(x)$ denote the number of k-*occurrences* of point x, i.e., the number of times x occurs in k-nearest-neighbor lists of other points from D. As the dimensionality of data increases, the distribution of k-occurrences becomes considerably skewed [16]. As a consequence, some data points, which we will refer to as *hubs*, are included in many more k-nearest-neighbor lists than other points. Moreover, in the rest of the text we will refer to the number of k-occurrences of point $x \in D$ as its *hubness score*. It has been shown that hubness appears in high-dimensional data as an inherent property of high dimensionality, and is not an artefact of finite samples nor a peculiarity of some specific data sets [16].

3.1 The Emergence of Hubs

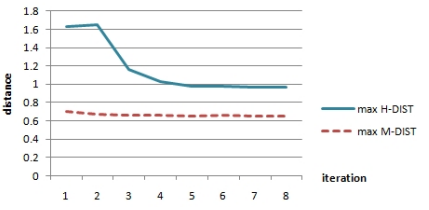
Hubness is closely related to the aforementioned concentration of distances in high-dimensional spaces. If distances do concentrate for a given data set, then its points are lying approximately on a hypersphere centered at the data mean. Naturally, if data is drawn from several distributions, as is usually the case in clustering problems, this could be rephrased by saying that data are lying approximately on several hyperspheres centered at the corresponding distribution means. However, it has been shown that the variance of distances to the mean is still non-negligible, regardless of the concentration phenomenon – for any finite number of dimensions [7]. This implies that some of the points will still end up being closer to the data (or cluster) mean than other points. It is well known that points closer to the mean tend to, on average, be closer to all other points, for any observed dimensionality. However, in high-dimensional data, this tendency is amplified [16]. On average, points which are closer to all other points will naturally have a higher probability of being included in k-nearest-neighbor lists of other points in the data set, which gives rise to an increase in their hubness scores.

3.2 Relation of Hubs to Data Clusters

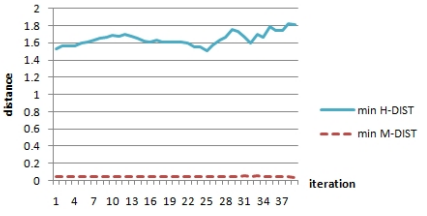
There has been some previous work on how well high-hubness elements cluster, as well as the general impact of hubness on clustering algorithms [16]. A correlation between low-hubness elements and outliers was also observed. A low hubness score indicates that a point is on average far from the rest of the points and hence probably an outlier. In high-dimensional spaces, however, low-hubness elements are expected to occur by the very nature of these spaces and data distributions. These data points will lead to an average increase in intra-cluster dissimilarity. It was also shown for several clustering algorithms that hubs do not cluster well compared to the rest of the points. This is due to the fact that some hubs are actually close to points in different clusters. Hence, they also lead to a decrease in inter-cluster dissimilarity. However, this does not necessarily hold for an arbitrary cluster configuration.



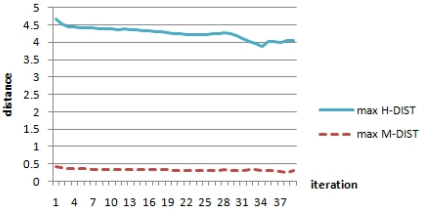
(a) Minimal distances for $d = 30$



(b) Maximal distances for $d = 30$



(c) Minimal distances for $d = 2$



(d) Maximal distances for $d = 2$

Fig. 1. Evolution of minimal and maximal distances from cluster centroids to hubs and medoids on synthetic data for neighborhood size 10 in case of 10 data clusters

It was already mentioned that points closer to cluster means tend to have higher hubness scores than the rest of the points. A natural question which arises is: *Are hubs medoids?* When observing the problem from the perspective of partitioning clustering approaches, of which K-means is the most commonly used representative, a similar question might also be posed: *Are hubs the closest points to data centroids in clustering iterations?* To answer this question, we ran K-means++ [17] multiple times on several randomly generated Gaussian mixtures for various fixed numbers of dimensions, observing the high-dimensional case. We measured in each iteration the distance from current cluster centroid to the medoid and to the hub, and scaled by the average intra-cluster distance. This was measured for every cluster in all the iterations, and for each iteration the minimal and maximal distance from any of the centroids to the corresponding hub and medoid were computed. Figure 1 gives example plots of how these ratios evolve through iterations for the case of 10-cluster data, used neighborhood size 10, with 30 dimensions for the high-dimensional case, and 2 dimensions to illustrate low-dimensional behavior.

It can be noticed from the charts that, in the low-dimensional case, hubs in the clusters are far away from the centroids, even farther than average points. There is no correlation between data means and high-hubness instances in the low-dimensional scenario. On the other hand, for the high-dimensional case, we observe that the minimal distance from centroid to hub converges to minimal distance from centroid to medoid. This implies that some medoids are in fact cluster hubs. Maximal distances to hubs and medoids, however, do not match. There exist hubs which are not medoids, and vice versa. Also, we observe that maximal distance to hubs also drops with iterations, hinting that as the iterations progress, centroids are becoming closer and closer to data hubs. This brings us to the idea that will be explained in detail in the following section: *Why not use hubs to approximate data centers?* After all, we expect points with high

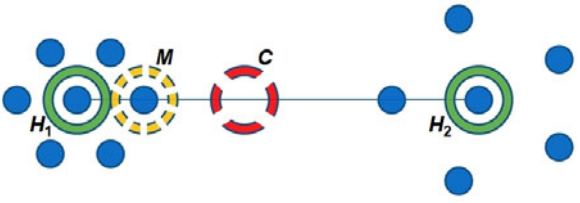


Fig. 2. Illustrative example. The red dashed circle marks the centroid (C), yellow dotted circle the medoid (M), and green circles denote two elements of highest hubness (H_1, H_2), for neighborhood size 3.

hubness scores to be closer to centers of relatively dense regions in high-dimensional spaces than the rest of the data points, making them viable candidates for representative cluster elements. We are not limited to observing only the points with the highest hubness scores, we can also take advantage of hubness information for any given data point. More generally, in case of irregularly shaped clusters, hubs are expected to be found near the centers of compact sub-clusters, which is also beneficial.

4 Hub-Based Clustering

If hubness is viewed as a kind of local centrality measure, it may be possible to use hubness for clustering in various ways. In order to test this hypothesis, we opted for an approach that allows observations about the quality of resulting clustering configurations to be related directly to the property of hubness, instead of being a consequence of some other attribute of the clustering algorithm. Since it is expected of hubs to be located near the centers of compact sub-clusters in high-dimensional data, a natural way to test the feasibility of using them to approximate these centers is to compare the hub-based approach with some centroid-based technique. For that reason, the considered algorithms are made to resemble K-means, by being iterative approaches for defining clusters around separated high-hubness data elements.

As Fig. 1 showed, centroids and medoids in K-means iterations tend to converge to locations close to high-hubness points. This implies that using hubs instead of either of these could actually speed up the convergence of the algorithms leading it straight to the promising regions in the data space. To illustrate this point, consider the simple example shown in Fig. 2, which mimics in two dimensions what normally happens in multidimensional data, and suggests that not only might taking hubs as centers in following iterations provide quicker convergence, but that it also might prove helpful in finding the best end configuration. Centroids depend on all current cluster elements, while hubs depend mostly on their neighboring elements and therefore carry *local* centrality information. We will consider two types of hubness below, namely *global* hubness and *local* hubness. We define local hubness as a restriction of global hubness on any given cluster, considered in the context of the current algorithm iteration. Hence, the local hubness score represents the number of k-occurrences of a point in k-nearest-neighbor lists of elements from within the same cluster.[1]

[1] Henceforth, we will use the capitalized K to represent the desired number of clusters and small k for neighborhood size.

The fact that hubs emerge close to centers of dense subregions might suggest some sort of a relationship between hubness and the density estimate at the observed data point. There are, however, some important differences. First of all, hubness does not depend on scale. Let D_1 and D_2 be two separate sets of points. If the local distance matrices defined on each of them separately are proportional, we might think of D_1 and D_2 as two copies of the same abstract data model appearing at different scales. Even though the density estimate might be significantly different, depending on the defining volumes which are affected by scale, there will be a perfect match in hubness scores of the corresponding points. However, there is a more subtle difference. Let $D_k(x)$ be the set of points where x is among the k nearest neighbors. Hence, the hubness score of x is then given by $N_k(x) = |D_k(x)|$. For each $x_i \in D_k(x)$, whether point x is among the k nearest neighbors of x_i depends on two things: $distance(x, x_i)$, and the density estimate at point x_i, not the density estimate at point x. Consequently, a hub might be a k-neighbor for points where density is high, as well as for points where density is low. Therefore, there is no direct correspondence between the magnitude of hubness and point density. Naturally, since hubs tend to be close to many points, it would be expected that density estimates at hub points are not low, but they do not necessarily correspond to the points of highest density among the data. Also, in order to calculate the exact volume of the neighborhood around a given point, one needs to have a suitable data representation. For hubness, one only needs the distance matrix.

Computational complexity of hubness-based algorithms is mostly determined by the cost of computing hubness scores. Computing the entire distance matrix may not be feasible for some very large datasets. However, it was demonstrated in [18] that it is possible to construct a k-NN graph (from which hubness scores can be read) in $\Theta(ndt)$, where the user-defined value $t > 1$ expresses the desired quality of graph construction. It was shown that good quality may be achieved with small values of t.

4.1 Deterministic Approach

A simple way to employ hubs for clustering is to use them as one would normally use centroids. Also, it allows us to make a direct comparison with the K-means algorithm. The algorithm, referred to as *K-hubs*, is given in Algorithm 1.

Algorithm 1 K-hubs

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
repeat
    for all Cluster c ∈ clusters do
        DataPoint h = findClusterHub(c);
        setClusterCenter(c, h);
    end for
    clusters = formClusters();
until noReassignments
return clusters
```

After initial evaluation on synthetic data, it became clear that even though the algorithm manages to find good and even best configurations often, it is quite sensitive to initialization. To increase the probability of finding the global optimum, we resorted to the stochastic approach described in the following section. However, even though K-hubs exhibited low stability, it converges to the stable configurations very quickly, in no more than four iterations on all the data sets used for testing, most of which contained around 10000 data instances.

4.2 Probabilistic Approach

Even though points with highest hubness are without doubt the prime candidates for cluster centers, there is no need to disregard the information about hubness scores of other points in the data. In the algorithm described below, we implemented a squared hubness-proportional stochastic scheme based on the widely used simulated annealing approach to optimization [19]. The temperature factor was introduced to the algorithm, so that it may start as being entirely probabilistic and eventually end by executing deterministic K-hubs iterations. We will refer to this algorithm, specified by Algorithm 2, as *hubness-proportional clustering* (HPC).

Algorithm 2 HPC

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
float t = t_0; {initialize temperature}
repeat
    float θ = getProbFromSchedule(t);
    for all Cluster c ∈ clusters do
        float choice = randomFloat(0,1);
        if choice < θ then
            DataPoint h = findClusterHub(c);
            setClusterCenter(c, h);
        else
            for all DataPoint x ∈ c do
                setChoosingProbability(x, N_k^2(x));
            end for
            normalizeProbabilities();
            DataPoint h = chooseHubProbabilistically(c);
            setClusterCenter(c, h);
        end if
    end for
    clusters = formClusters();
    t = updateTemperature(t);
until noReassignments
return clusters
```

The reason why hubness-proportional clustering is reasonable in the context of high dimensionality lies in the skewness of the distribution of k-occurrences. Namely, there exist many more data points having a low hubness score, making them bad candidates

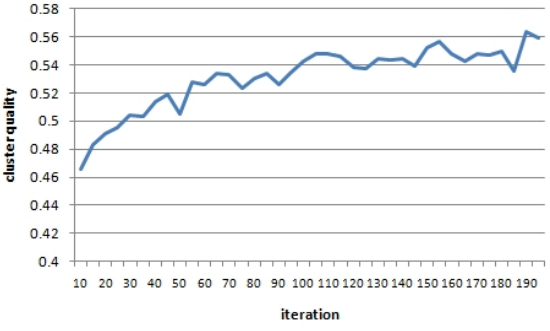


Fig. 3. Estimated quality of clustering for various durations of probabilistic search in HPC

for cluster centers. Such points will have a low probability of being selected. To further emphasize this, we use the square of the actual hubness score instead of making the probabilities directly proportional to $N_k(x)$.

The HPC algorithm defines a search through the data space based on hubness as a kind of a local centrality estimate. It is possible to take as the output the best solution according to some predefined criterion like minimum squared error, rather than simply taking the last produced cluster configuration. This may in some situations produce even better clustering results. We were mostly focused on finding the best stable hub configuration, thus we only used the last produced configuration to estimate the results for tests presented in the rest of the paper. To justify the use of the proposed stochastic scheme, we executed a series of initial tests for a synthetic mixture of Gaussians, for dimensionality $d = 50$, $n = 10000$ instances, and $K = 25$ clusters in the data. Neighborhood size was set to $k = 10$ and for each preset number of probabilistic iterations in the annealing schedule, the clustering was run 50 times, each time re-initializing the seeds. The results are displayed in Fig. 3. The silhouette index [20] was used to estimate the clustering quality. Due to the significant skewness of the squared hubness scores, adding more probabilistic iterations helps in achieving better clustering, up to a certain plateau that is eventually reached. The same shape of the curve also appears in the case of not taking the last, but the error-minimizing configuration.

5 Experiments and Evaluation

We tested our approach on various high-dimensional synthetic and real-world data sets. We will use the following abbreviations in the forthcoming discussion: KM (K-Means), GKH (Global K-Hubs), LKH (Local K-Hubs), GHPC (Global Hubness-Proportional Clustering) and LHPC (Local Hubness-Proportional Clustering), *local* and *global* referring to the type of hubness score that was used (see Section 4). For all algorithms, including KM, we used the D^2 initialization procedure described in [17]. Hubness could also be used for cluster initialization, an option which we have not fully explored yet. For determining $N_k(x)$ we used $k = 10$ by default in our experiments involving synthetic data, since the generated sets were large enough to insure that such a value of k would not overly smooth out hubness. There is no known way of selecting the best k for

finding neighbor-sets, with the problem also depending on the particular application. To check how the choice of k reflects on hubness-based clustering, we ran a series of tests on a fixed synthetic data set for a range of k values. The results suggest that the algorithms proposed in this paper are not very sensitive to changes of k, with no observable monotonicity in scores, meaning that the clustering quality does not rise with rising k, or vice versa. (We omit these charts due to space considerations.)

In the following sections, as a baseline we will use K-means++, since it is suitable for determining the feasibility of using hubness to estimate local centrality of points.

5.1 Synthetic Data: Gaussian Mixtures

For comparing the resulting clustering quality, we used mainly the silhouette index as an unsupervised measure of configuration validity, and average cluster entropy as a supervised measure of clustering homogeneity. Since most of the generated data sets are "solvable," i.e., consist of non-overlapping Gaussian distributions, we also report the normalized frequency with which the algorithms were able to find these perfect configurations. We ran two lines of experiments, one using 5 Gaussian generators, the other using 10. For each of these, we generated data of ten different high dimensionalities, more specifically for $10, 20, 30, \ldots, 100$. In each case, 10 different Gaussian mixtures were randomly generated, resulting in 200 different generic sets, 100 of them containing 5 data clusters, the other containing 10. On each of the data sets, KM and all of the hub-based algorithms have been executed 30 times and the averages were calculated.

Table 1 shows the final summary of all these runs. (Henceforth, we use boldface to denote measurements that are significantly better than others, in the sense of having no overlap of surrounding one-standard deviation intervals.) Global hubness is definitely to be preferred, especially in the presence of more clusters, which further restricts neighbor sets in the case of local hubness scores. Probabilistic approaches significantly outperform the deterministic ones, even though GKH and LKH also sometimes converge to the best configurations, but much less frequently. More importantly, the best overall algorithm in these tests was GHPC, which outperformed KM on all basis, having lower average entropy, a higher silhouette index, and a much higher frequency of finding the perfect configuration. This suggests that GHPC is a good option for clustering high-dimensional Gaussian mixtures. Regarding the number of dimensions when the actual

Table 1. Averaged results of algorithm runs on high-dimensional mixtures of Gaussians. Standard deviations taken over dataset averages are given on the side.

		LKH	GKH	LHPC	GHPC	KM
$K = 5$	Silhouette	0.47 ± 0.03	0.52 ± 0.02	$\mathbf{0.62} \pm 0.02$	$\mathbf{0.63} \pm 0.02$	0.58 ± 0.02
	Entropy	0.31 ± 0.04	0.16 ± 0.01	0.078 ± 0.02	$\mathbf{0.05} \pm 0.01$	0.10 ± 0.01
	Perf.	0.35 ± 0.05	0.41 ± 0.06	$\mathbf{0.78} \pm 0.08$	$\mathbf{0.78} \pm 0.06$	0.57 ± 0.05
$K = 10$	Silhouette	0.39 ± 0.03	0.49 ± 0.01	0.53 ± 0.02	$\mathbf{0.59} \pm 0.01$	0.54 ± 0.01
	Entropy	0.51 ± 0.06	0.21 ± 0.01	0.21 ± 0.03	$\mathbf{0.07} \pm 0.01$	0.12 ± 0.01
	Perf.	0.07 ± 0.03	0.07 ± 0.03	0.32 ± 0.06	$\mathbf{0.42} \pm 0.07$	0.14 ± 0.02

Table 2. Estimated cluster quality at various noise levels

	GKH		GHPC		KM	
	Silhouette	Entropy	Silhouette	Entropy	Silhouette	Entropy
Avg. total	0.78	0.35	**0.83**	**0.28**	0.70	0.62
Avg. noise 10–50%	0.77	0.37	**0.82**	**0.29**	0.68	0.66
Avg. noise 30–50%	0.73	0.42	**0.80**	**0.32**	0.66	0.71

improvements begin to show, in our lower-dimensional test runs, GHPC was better already on 6-dimensional mixtures. Since we concluded that using global hubness leads to better results, we only consider GKH and GHPC in the rest of the experiments.

5.2 Clustering in the Presence of High Noise Levels

Real-world data often contains noisy or erroneous values due to the nature of the data-collecting process. It is natural to assume that hub-based algorithms will be more robust with respect to noise, since the hubness-proportional search is driven mostly by the highest-hubness elements, not the outliers. In the case of KM, all of the instances within the current cluster directly determine the location of the centroid in the next iteration. When the noise level is low, some sort of outlier removal technique may be applied. In setups involving high levels of noise this is not the case. We generated a data set of 10000 instances as a mixture of 5 clearly separated Gaussians, farther away from each other than in the previously described experiments. To this data we incrementally added noise, 250 instances at a time, drawn from a uniform distribution on a hypercube containing all the data points. In other words, clusters were immersed in uniform noise. The highest level of noise for which we tested was the case when there was an equal number of actual data instances in original clusters and noisy instances. At each noise level, KM, GKH and GHPC were run 50 times each. To reduce the influence of noise on hubness estimates, $k = 20$ was used. The silhouette index and average entropy were computed only on the non-noisy restriction of the data, i.e., the original Gaussian clusters. A brief summary of total averages is given in Table 2. The hub-based algorithms show substantial improvements in higher noise levels, which is a useful property. The difference in entropy was quite convincing, 0.62 for KM and only 0.28 for GHPC on average over all the runs. Even though KM had a smaller square error calculated on the combined noisy data set, hub-based approaches were better at finding the underlying structure of the original data.

5.3 Experiments on Real-World Data

The two-part Miss-America data set (`cs.joensuu.fi/sipu/datasets/`) was used for evaluation. Each part consists of 6480 instances having 16 dimensions. Results were compared for various predefined numbers of clusters in algorithm calls. Each algorithm was tested 50 times for each number of clusters. Neighborhood size was set to 5. The silhouette index was again used to measure quality. For all the experiments on real-world data we used only the silhouette index because categories in real world data sets often violate the cluster assumption, so any conclusions based on label entropy would

Table 3. Cluster configuration quality measured by the silhouette index, on the Miss-America data set, parts I and II, for various cluster numbers

	K	2	4	6	8	10	12	14	16
Part I	GKH	0.40	0.20	0.10	0.09	0.06	0.06	0.06	0.06
	GHPC	**0.42**	**0.31**	**0.31**	**0.21**	**0.21**	**0.15**	**0.14**	**0.12**
	KM	0.21	0.13	0.12	0.08	0.08	0.08	0.07	0.07
Part II	GKH	0.31	0.13	0.07	0.06	0.05	0.05	0.05	0.05
	GHPC	**0.36**	**0.23**	0.12	0.09	0.09	0.08	0.09	0.07
	KM	0.18	0.12	0.10	0.08	0.08	0.08	0.08	0.07

Table 4. Cluster configuration quality measured by the silhouette index, on some UCI datasets

dataset	size	d	K	GKH-Sil.	GHPC-Sil.	KM-Sil.
wdbc	569	30	2	0.43	0.43	0.43
spambase	4601	57	2	0.28	**0.44**	0.39
arcene	100	1000	2	0.35	0.36	0.34
ovarian	253	15154	2	0.22	0.22	0.21
iris	158	4	3	0.56	0.58	0.58

be less reliable. The results for both parts of the data set are given in Table 3. GHPC clearly outperformed both other algorithms, showing highest improvements for smaller numbers of clusters. Observe that for $K = 2$ it achieved a double of KM's silhouette index, 0.42 compared to 0.21 on Part I and 0.36 compared to 0.18 on Part II.

Tests were also run on several UCI datasets (`archive.ics.uci.edu/ml/datasets.html`). Values of all the individual features in the data sets were normalized prior to testing. The results, shown in Table 4, are mostly comparable between the algorithms. Value of k was set to 20. The datasets were simple, composed only of few clusters, so the results being similar is not surprising. Note, however, that GHPC did as well as KM on *Iris* dataset, which is only 4-dimensional. This suggests that hubness-based algorithms might also be successfully applied in some lower-dimensional cases.

6 Conclusions and Future Work

Using hubness for data clustering has not previously been attempted. We have shown that using hubs to approximate local data centers is not only a feasible option, but also frequently leads to improvement over the centroid-based approach. In our experiments GHPC (Global Hubness-Proportional Clustering) had an overall best performance in various test settings, on both synthetic and real-world data, as well as in the presence of high levels of artificially introduced noise. Global hubness estimates are generally to be preferred to the local ones if used in the proposed framework. Hub-based algorithms are designed specifically for high-dimensional data. This is an unusual property, since

the performance of most standard clustering algorithms deteriorates with an increase of dimensionality. Hubness, on the other hand, is an inherent property of high-dimensional data, and this is precisely where GHPC may offer greatest improvement.

The proposed algorithms represent only one possible approach to using hubness for improving high-dimensional data clustering. Next, we will explore related agglomerative approaches. However, even the described algorithms offer space for improvements, since some questions are left unanswered: What is the best choice of k? Is it possible to automatically determine the appropriate number of clusters by carefully inspecting the hubs? In cases such as one depicted in Fig. 2 that would probably be possible. What is the best annealing schedule in GHPC? Is it possible to use several different values of k in LHPC to avoid over-smoothing the hubness estimates for small clusters over iterations and make local hubness more useful? Is there a better way to initialize hubness-based algorithms? We plan to address all these details in our future work.

Acknowledgments. This work was supported by the Slovenian Research Agency program Knowledge Technologies P2-0103, and the Serbian Ministry of Science and Technological Development project no. OI174023.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
2. Aggarwal, C.C., Yu, P.S.: Finding generalized projected clusters in high dimensional spaces. In: Proc. 26th ACM SIGMOD Int. Conf. on Management of Data, pp. 70–81 (2000)
3. Kailing, K., Kriegel, H.P., Kröger, P., Wanka, S.: Ranking interesting subspaces for clustering high dimensional data. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 241–252. Springer, Heidelberg (2003)
4. Kailing, K., Kriegel, H.P., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proc. 4th SIAM Int. Conf. on Data Mining (SDM), pp. 246–257 (2004)
5. Kriegel, H.P., Kröger, P., Renz, M., Wurst, S.: A generic framework for efficient subspace clustering of high-dimensional data. In: Proc. 5th IEEE Int. Conf. on Data Mining (ICDM), pp. 250–257 (2005)
6. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional space. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 420–434. Springer, Heidelberg (2000)
7. François, D., Wertz, V., Verleysen, M.: The concentration of fractional distances. IEEE Transactions on Knowledge and Data Engineering 19(7), 873–886 (2007)
8. Durrant, R.J., Kabán, A.: When is 'nearest neighbour' meaningful: A converse theorem and implications. Journal of Complexity 25(4), 385–397 (2009)
9. Agirre, E., Martínez, D., de Lacalle, O.L., Soroa, A.: Two graph-based algorithms for state-of-the-art WSD. In: Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP), pp. 585–593 (2006)
10. Tran, T.N., Wehrens, R., Buydens, L.M.C.: Knn density-based clustering for high dimensional multispectral images. In: Proc. 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas Workshop, pp. 147–151 (2003)
11. Biçici, E., Yuret, D.: Locally scaled density based clustering. In: Beliczynski, B., Dzielinski, A., Iwanowski, M., Ribeiro, B. (eds.) ICANNGA 2007, Part I. LNCS, vol. 4431, pp. 739–748. Springer, Heidelberg (2007)

12. Zhang, C., Zhang, X., Zhang, M.Q., Li, Y.: Neighbor number, valley seeking and clustering. Pattern Recognition Letters 28(2), 173–180 (2007)
13. Hader, S., Hamprecht, F.A.: Efficient density clustering using basin spanning trees. In: Proc. 26th Annual Conf. of the Gesellschaft für Klassifikation, pp. 39–48 (2003)
14. Ding, C., He, X.: K-nearest-neighbor consistency in data clustering: Incorporating local information into global optimization. In: Proc. ACM Symposium on Applied Computing (SAC), pp. 584–589 (2004)
15. Chang, C.T., Lai, J.Z.C., Jeng, M.D.: Fast agglomerative clustering using information of k-nearest neighbors. Pattern Recognition 43(12), 3958–3968 (2010)
16. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research 11, 2487–2531 (2010)
17. Arthur, D., Vassilvitskii, S.: k-means++: The advantages of careful seeding. In: Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1027–1035 (2007)
18. Chen, J., Fang, H., Saad, Y.: Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. Journal of Machine Learning Research 10, 1989–2012 (2009)
19. Corne, D., Dorigo, M., Glover, F.: New Ideas in Optimization. McGraw-Hill, New York (1999)
20. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley, Reading (2005)

Spatial Entropy-Based Clustering for Mining Data with Spatial Correlation

Baijie Wang and Xin Wang

Department of Geomatics Engineering, University of Calgary, 2500 University Drive, Calgary, AB, Canada
{baiwang,xcwang}@ucalgary.ca

Abstract. Due to the inherent characteristics of spatial datasets, spatial clustering methods need to consider spatial attributes, non-spatial attributes and spatial correlation among non-spatial attributes across space. However, most existing spatial clustering methods ignore spatial correlation, considering spatial and non-spatial attributes independently. In this paper, we first prove that spatial entropy is a monotonic decreasing function for non-spatial attribute similarity and spatial correlation. Then we propose a novel density-based spatial clustering method called SEClu, which applies spatial entropy in measuring non-spatial attribute similarity and spatial correlation during the clustering process. The experimental results from both the synthetic data and the real application demonstrate that SEClu can effectively identify spatial clusters with spatial correlated patterns.

Keywords: Spatial Clustering, Spatial Entropy, Spatial Correlation.

1 Introduction

Spatial clustering is an active research area in the field of spatial data mining, which groups objects into meaningful subclasses based on their spatial and non-spatial attributes [1], [2]. In spatial data, spatial attributes, such as coordinates, describe the locations of objects. Non-spatial attributes include the non-spatial features of objects, such as oil saturation, population or species. Meanwhile, spatial correlation generally exists in spatial datasets, describing the dependent relationship on the non-spatial attributes across space.

Spatial clustering has previously been based on only the spatial attributes of the data. However, the non-spatial attributes may have a significant influence on the results of the clustering process. For example, in image processing, the general procedure for region-based segmentation compares a pixel with its immediate surrounding neighbors. Region growing is heavily based on not only on the location of pixels but also on the attributes of those pixels [3].

Meanwhile, spatial correlation is always considered important for spatial datasets. Spatial correlation always indicates the dependency between the spatial and non-spatial attributes, with the chance that some cause and effect lead to it. Of particular interest the higher degrees of spatial correlation [2].

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 196–208, 2011.

In order to identify meaningful clusters from spatial datasets, spatial clustering methods need to consider spatial attributes, non-spatial attributes and inherent spatial correlations during the clustering process. We will illustrate these requirements using the following examples. Fig. 1 (a) includes two round shaped groups. The grey values of the left group change to darker consistently from the center, which shows strong spatial correlation. The grey values of the right group are randomly distributed, indicating weak or no spatial correlation meaning that this group should not be identified as a cluster. Similarly, we should be able to identify an arbitrary shaped cluster with spatial correlation such as the cluster shown in Fig. 1 (b). Fig. 1 (c) shows a raster data set with intensity as the non-spatial attributes. Since the data objects are evenly distributed (i.e. cell by cell), without considering the non-spatial attribute, the whole dataset might be clustered into one. However, the values of the non-spatial attribute form two spatially correlated clusters with the non-spatial attribute values differing significantly at the boundary. Thus, two spatial clusters should be identified from the spatial correlated area in this dataset.

However, most existing methods only focus on the spatial attributes or consider spatial and non-spatial independently. These methods are not suitable for those spatial datasets in which the non-spatial attributes and spatial correlation play important roles.

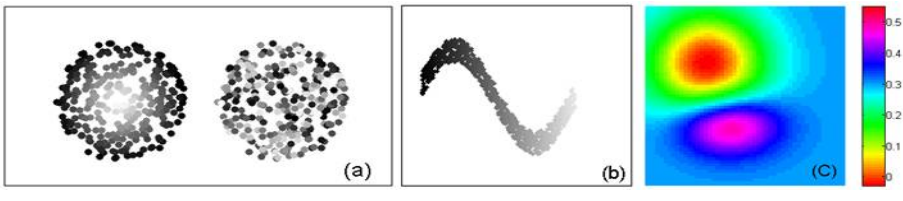


Fig. 1. Three sample spatial datasets. (c) is with color in electronic version.

Spatial entropy is the extension of Shannon Entropy with the spatial configuration. It measures the distribution of a non-spatial attribute in the spatial domain [4]. In this paper, we propose to apply spatial entropy to measure local non-spatial attribute similarity and spatial correlation. A novel spatial entropy-based clustering method, called SEClu, is proposed to take into account spatial attributes, non-spatial attributes and spatial correlations.

Spatial clustering based on both non-spatial attributes and spatial correlation is a new research topic. To our best knowledge, none of the previous methods are able to identify useful clusters directly from spatial correlation, hence the focus of this research. The main contributions of this paper are summarized below.

First, we establish that spatial entropy is an unbiased measure of local non-spatial attribute similarity and spatial correlation. Specifically, spatial entropy decreases for both local non-spatial attribute similarity and spatial correlation, thus a smaller spatial entropy value denotes that the corresponding spatial data is more likely to be grouped into a cluster.

Second, we propose a novel spatial clustering method named spatial entropy-based clustering, SEClu. SEClu discovers clusters that are not only dense spatially but that also have high spatial correlation based on their non-spatial attributes across that space.

Third, SEClu is demonstrated using both synthetic data and data from a real application. Results show that SEClu discovers more meaningful spatial clusters than methods considering spatial and non-spatial attributes independently.

The remainder of the paper is organized as follows. In Section 2, we present some related work. Section 3 introduces spatial entropy as the measure of spatial correlation and proposes a new density-based spatial clustering algorithm termed SECLu. Experiments on synthetic and real datasets are performed in Section 4. Section 5 summarizes the paper and discusses priorities for future work.

2 Related Work

In this section, we review some related spatial clustering methods.

DBSCAN is the first proposed density-based spatial clustering method. It starts from an arbitrary point q by performing a neighborhood query. If the neighborhood contains fewer than *MinPts* points, then point q is labeled as noise. Otherwise, a cluster is created and all points in q's neighborhood are placed in this cluster. Then the neighborhood of each of q's neighbors is examined to see if it can be added to the cluster. If so, the process is repeated for every point in this neighborhood, and so on. If a cluster cannot be expanded further, DBSCAN chooses another arbitrary unlabelled point and repeats the process. Although DBSCAN gives extremely good results and is efficient in many datasets, it is not suitable for cases where the non-spatial attributes play a role in determining the desired clusters since it does not take into consideration the non-spatial attributes in the dataset [1],[3].

Over the years, very few algorithms have been proposed for dealing with both spatial and non-spatial attributes during clustering process. One option is to handle non-spatial attributes and spatial attributes in two separate steps, as described in CLARANS [5]. The other option is to deal with the non-spatial attributes and spatial attributes together in the clustering process. The similarity functions for non-spatial attributes, and distance functions for spatial attributes, are handled simultaneously in order to define the overall similarity between objects. Algorithms that have taken this approach include GDBSCAN [1], DBRS [3], and Clustering of Multi-represented Objects [6].

GDBSCAN [1] takes into account the non-spatial attributes of an object as a "weight" attribute, which is defined by the weighted cardinality of the singleton containing the object. The weight can be the size of the area of the clustering object, or a calculated value from several non-spatial attributes. DBRS [3] introduces the concept of 'purity' to determine the categorical attributes of objects in the neighborhood. 'Purity' is defined as the percentage of objects in the neighborhood, with the same characteristic for a particular non-spatial attribute as the center object. For non-spatial attributes, this can avoid creating clusters

of points with different values, even though these points may be close to one other. However, 'purity' is only defined for categorical non-spatial attributes. Clustering of Multi-represented Objects (CMRO) [6] extends DBSCAN by retrieving information from one attribute to multiple attributes. Within a set of attributes, either spatial or non-spatial, 'density reachability' is defined as the union or intersection of the selected attributes.

Even though all of the above methods consider the significance of non-spatial attributes, they ignore the spatial correlations between the spatial and non-spatial attributes.

3 Spatial Entropy-Based Clustering

In this paper, we are interested in identifying arbitrary-shaped clusters based on spatial correlations. Since non-spatial attributes in a spatially correlated cluster usually change continuously, the values of non-spatial attributes might differ significantly for the whole cluster. Therefore, in this research, we require similarity in a small area but not in the whole cluster.

In the following, we will first introduce spatial entropy and then justify that it is an unbiased measure in spatial clustering with respect to local non-spatial similarity and spatial correlation.

3.1 Spatial Entropy

Spatial entropy is an information measure of non-spatial attributes that also takes into account the influence of spatial spaces. Various forms of spatial entropy have been developed for how to quantify the extent of the role played by space [4],[7]. In this paper, we select the one from [4] because it is simple and can handle both discrete and continuous non-spatial attributes.

$$d_i^{int} = \begin{cases} \frac{1}{|D_i| \times |D_i - 1|} \sum_{j=1, j \in D_i}^{|D_i|} \sum_{k=1, k \neq j, k \in D_i}^{|D_i|} dist(j,k) & \text{if} |D_i| > 1 \\ \lambda & \text{otherwise} \end{cases} \quad (1)$$

$$d_i^{ext} = \begin{cases} \frac{1}{|D_i| \times |D - D_i|} \sum_{j=1, j \in D_i}^{|D_i|} \sum_{k=1, k \neq j, k \notin D_i}^{|D - D_i|} dist(j,k) & \text{if} D \neq D_i \\ \beta & \text{otherwise} \end{cases} \quad (2)$$

Given a dataset D with a non-spatial attribute *prop* in spatial spaces $\{S_1 \cdots S_m\}$, $\{D_i \cdots D_i \cdots D_n\}$ is a partition of D based on *prop*, i.e. $D_i \subset D$, $\cup D_i = D$, and $D_i \cap D_j = \oslash$, $i \neq j$. $p_1, \cdots, p_i, \cdots, p_n$ are the fraction of the number of objects in category D_i over the whole dataset D, i.e. $p_i = |D_i|/|D|$ and $\sum p_i = 1$. The intra-distance of D_i, denoted by d_i^{int} is the average distance between objects in D_i (shown in Eq. (1)). The extra-distance of D_i, denoted by d_i^{ext} is the average distance of objects in D_i to other partition classes of D (shown in Eq. (2)).

In Eq. (1), when D_i is empty or contains only one object, we assume its intra-distance is very small and a small constant λ is assigned to d_i^{int} to avoid the influence of null values on the computation. In Eq. (2), when D_i includes all of the objects in D, i.e. all objects have similar values of *prop*, we assume that the extra-distance d_i^{ext} is very large, and assign the extra-distance with a large constant β. $dist(j,k)$ is the distance between objects j and k in spatial spaces.

Definition 1. *The* ***spatial entropy*** *of dataset D based on its partition $\{D_1, \cdots, D_i, \cdots, D_n\}$ is defined as* (from [4])*:*

$$H_s(p_1 \cdots p_i \cdots p_n) = -\sum_{i=1}^{n} \frac{d_i^{int}}{d_i^{ext}} p_i \log_2(p_i) \tag{3}$$

In this definition, a spatial configuration d_i^{int}/d_i^{ext} is added as a weight factor in the Shannon Entropy. The weight factor decreases when either the intra-distance decreases or the extra-distance increases, which enables spatial entropy to measure the spatial distribution. Besides, given D and its partition, the spatial entropy is similar to Shannon Entropy in that it reaches the maximum value when $p_1 = \cdots p_i \cdots = p_n$.

3.2 Using Spatial Entropy in Spatial Clustering

In this section, we demonstrate that spatial entropy is a monotonic decreasing function for local non-spatial attribute similarity and spatial correlation.

Spatial Entropy vs. Local Non-Spatial Similarity. The non-spatial attribute *prop* of the spatial dataset D can be viewed as a random variable with its probability density function approximated using a histogram. If the non-spatial attribute *prop* is random, it follows an even distribution. As the local non-spatial similarity increases, *prop* tends to be more concentrated.

It has been shown that Shannon entropy of an even distribution reaches maximum value and tends to decrease as the concentration of the distribution increases. Spatial Entropy H_s is a special form of Shannon entropy and has a spatial configuration weight factor d_i^{int}/d_i^{ext}. Even though each object's non-spatial attribute is correlated within the spatial spaces, the probability distribution of *prop* is independent from d_i^{int}/d_i^{ext}. Hence, the weight factor d_i^{int}/d_i^{ext} does not influence the property of spatial entropy H_s, which is a measure of randomness. Therefore, when *prop* follows an even distribution, its spatial entropy value reaches the maximum, otherwise the spatial entropy H_s decreases as the concentration of the probability distribution increases.

In Fig. 2 (a), datasets 1 and 2 have the same spatial attributes, but points in dataset 2 have more similar non-spatial attributes than dataset 1. From the histograms, it is evident that more than 60% of points in dataset 2 have grey values between [150,200] while values in dataset 1 are random. Fig. 2 (b) shows that the spatial entropy value decreases from dataset 1 to dataset 2.

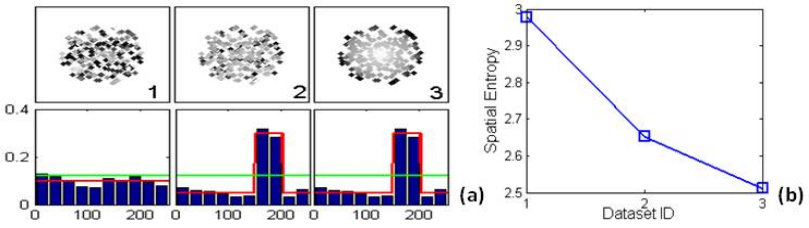


Fig. 2. (a) Scatter plots and histograms for three grey value point datasets (b) Spatial entropy for the three datasets shown in Fig. 2 (a)

Spatial Entropy vs. Spatial Correlation. Spatial entropy measures spatial correlation by quantifying spatial diversity. As in [8], the First Law of Geography states " everything is related to everything else, but near things are more related than distant things". It implies not only the general existence of spatial correlation but also, as in [4], spatial diversity increases when either the distance between different objects decreases or the distance between similar entities increases. The intra-distance and the extra-distance are integrated into the spatial entropy using the form of d_i^{int}/d_i^{ext}, which keeps the spatial entropy decreasing when similar objects are close and diverse objects are far from each other. Spatial objects where similar non-spatial attribute values are close and where spatial objects with different non-spatial attributes are far from each other, d_i^{int}/d_i^{ext} decreases. Therefore, spatial entropy decreases when spatial correlation increases.

In Fig. 2 (a), datasets 2 and 3 have the same spatial and non-spatial attributes, but different distributions for the non-spatial attribute. Here the spatial correlation increases from 2 to 3 (high value points centered and low value in the periphery), and the spatial entropy value decreases accordingly, as shown in Fig. 2 (b).

3.3 A Spatial Entropy-Based Spatial Clustering Method

In this section, we propose a novel spatial clustering method, Spatial Entropy-based Clustering (SEClu). Given a spatial dataset *SD* with a non-spatial attribute *prop*, a symmetric distance function *dist*, and parameters *Eps*, *MinNum* and *MaxSp*, we introduce the following definitions for SEClu.

Definition 2. *The* ***neighborhood*** *of spatial object p, denoted by* $N_{Eps}(p)$*, is defined as* $N_{Eps}(p) = \{q \in SD | dist(p,q) \leq Eps\}$ (from [9]).

SEClu extends DBSCAN by applying spatial entropy to control the local non-spatial similarity and spatial correlation of $N_{Eps}(p)$. The previous discussion demonstrates that the spatial entropy value decreases for the local non-spatial attribute similarity and that spatial correlation increases. Therefore, SEClu introduces the maximum threshold of spatial entropy, denoted by *MaxSp*.

In SEClu, a ***core object*** is an object whose neighborhood (1) has at least *MinNum* neighbors in spatial spaces, (2) has similar non-spatial attributes and

high spatial correlation in its neighborhood satisfying $H_s(N_{Eps}(p)) \leqslant MaxSp$. A ***border object*** is a neighbor object of a core object which is not a core object itself. Objects other than core objects or border objects are ***noise***.

Definition 3. *A spatial object p is* ***directly density-spEntropy-reachable*** *to an object q w.r.t Eps, MinNum, MaxSp if* (1) $q \in N_{Eps}(p)$; (2) $N_{Eps}(p) \geq MinNum$; *and* (3) $H_s(N_{Eps}(p)) \leq MaxSp$.

In the above definition, the second condition examines the density of the neighborhood of p. The third condition examines non-spatial attribute of the neighborhood of p. A smaller value of spatial entropy H_s implies objects in $N_{Eps}(p)$ have higher non-spatial similarity and spatial correlation.

Definition 4. *Spatial objects p and q are* ***density-spEntropy reachable*** (***DSR-reachable***) *w.r.t Eps, MinNum, MaxSp, denoted by DSR(p,q), if there is a chain of objects* $p_1 \cdots p_n$, $p_1 = q$, $p_n = p$ *such that* p_{i+1} *is directly density-spEntropy reachable from* p_i.

Definition 5. *A* ***density-spEntropy based cluster*** *C is a non-empty subset of SD satisfying:* $\forall p, q \in SD$, *if* $p \in C$ *and DSR(p,q) holds, then* $q \in C$.

It is obvious that for each pair of objects $(p, q) \in C$, when C is a density-spEntropy based cluster, $DSR(p, q)$ holds. Therefore, SEClu finds a cluster by identifying all objects that are density-spEntropy reachable.

Algorithm 1. SEClu(SD, Eps, $MinNum$, $MaxSp$)

```
1: for each unclassified p ∈ SD do
2:   if |N_Eps(p)| < MinNum or H_s(N_Eps(p)) > MaxSp then
3:     mark p as noise
4:   else
5:     creat a new cluster C and put x ∈ N_Eps(p) in C
6:     add x ∈ N_Eps(p) into a queue Q
7:     while Q is not empty do
8:       q = first object in Q and remove q from Q
9:       if |N_Eps(q)| ⩾ MinNum and H_s(N_Eps(q)) ⩽ MaxSp then
10:        for each object t ∈ N_Eps(q) do
11:          if t is unclassified then
12:            add t into Q and put t in C
13:          else if t is noise then
14:            put t into C
```

As shown in Algorithm 1, SEClu starts by querying the neighborhood of an arbitrary object p in spatial space to see if it is dense enough $N_{Eps}(p) \geq MinNum$. If not, p is labeled as noise, otherwise SEClu continues to check the non-spatial attribute. If the non-spatial attribute of p's neighborhood has a random pattern, i.e., it cannot satisfy $H_s(N_{Eps}(p)) \leq MaxSp$, then p is labeled as noise. Otherwise, a new cluster C is created and all objects $x \in N_{Eps}(p)$ are

placed in C. The neighborhood of each of p's neighbors is examined in the same way to see if it can be added to C. This process is repeated until all objects that are density-spEntropy reachable to p have been added to cluster C. If cluster C cannot be expanded further, SEClu chooses another unlabelled object and repeats this process until all objects have been assigned to a cluster or labeled as noise. The average complexity of SEClu is $O(n(\log n + k^2))$, where n is the number of the objects in SD and k is the average number of objects in $N_{Eps}(p_i)$.

Calculating Spatial Entropy Efficiently. In SEClu, the spatial entropy H_s is computed on the non-spatial attribute of $N_{Eps}(p)$. To be able to use H_s to measure the spatial correlation, a partition process of $N_{Eps}(p)$ is generated in the first step. Given a spatial dataset $D = N_{Eps}(p)$ with the non-spatial attribute *prop*, if *prop* is discrete it is binned into n slots with different values. If *prop* is continuous, it is binned into n contiguous slots $(\chi_1, \cdots, \chi_i, \cdots, \chi_n)$ with the interval of Δ. Then, each object in D is assigned to a unique slot based on its *prop* value, and a partition of D, denoted by $\{D_1, \cdots, D_i, \cdots, D_n\}$, is generated.

The number of subsets n should be selected carefully. If n is too large, each subset may contain a very small number of data and also result in a high computational cost. Sturges' rule [10] is widely recommended for choosing a histogram interval since it provides a good approximation of the best n in capturing the distribution pattern. In this paper, we adopt the Sturges' rule, and the subset number n is given by Eq. (4). Since SEClu is a density-based method, an effective way is to assign *MinNum* to N.

$$n = 1 + \log_2 N, \quad N \text{ is the number of objects in } D \tag{4}$$

In practice, spatial entropy H_s is computed numerically. Since prior knowledge of the distribution of *prop* is always unknown, p_i is estimated from the frequency $p_i = |D_i|/|D|$. d_i^{int} and d_i^{ext} can be computed from Eqs. (1) and (2), respectively. Assume D_i contains k objects, Eq. (1) calculates *dist* for $k(k-1)$ times, which computes the distance from each object to the other $(k-1)$ objects.

$$d_i^{int} = \begin{cases} \frac{2}{|D_i| \times |D_i - 1|} \sum_{j=1, j \in D_i}^{|D_i|} \sum_{k=j+1, k \in D_i}^{|D_i|} dist(j,k) & \text{if} |D_i| > 1 \\ \lambda & \text{otherwise} \end{cases} \tag{5}$$

To avoid the distance between each pair of objects in D_i being computed twice, Eq. (1) can be substituted to a low computation cost formula Eq. (5) when *dist* is a symmetric function. Eq. (5) does not calculate duplicate distances, which computes *dist* for $k(k-1)/2$ times, half from Eq. (1).

Also, H_s needs to be normalized in order to make a fair comparison. It has been demonstrated that $d_i^{int} \leq 2d_i^{ext}$ [7]. Also, for a discrete random variable its Shannon entropy value satisfies $0 \leq -\sum_{i=1}^{n} p_i \log_2(p_i) \leq \log_2 n$, then:

$$0 \leq H_s = -\sum_{i=1}^{n} \frac{d_i^{int}}{d_i^{ext}} p_i \log_2(p_i) \leq -2 \sum_{i=1}^{n} p_i \log_2(p_i) \leq 2 \log_2 n \tag{6}$$

Algorithm 2. Spatial Entropy $H_s(D)$

1: Bin D into $\{D_1 \cdots D_i \cdots D_n\}$ based on $D.prop$
2: **for** each D_i **do**
3: Compute p_i, d_i^{int} and d_i^{ext} (from Eqs (5) and (2), respectively)
4: compute H_s (from Eq. (3))
5: **return** H_s

In the following, all spatial entropy values are normalized by $H_s/2\log_2 n \in [0, 1]$. Algorithm 2 shows the pseudocode for the spatial entropy calculation.

Spatial Entropy Parameter *Maxsp*. In SEClu, the parameters *Eps* and *MinNum* can be determined by using the heuristic method in [9]. Besides, given *Eps* and *MinNum*, *MaxSp* can be determined by the following rationale. Meeting the requirement of $N_{Eps}(p) \geq MinNum$, p should form a core object if it satisfies $H_s(N_{Eps}(p)) \leq MaxSp$. Thus, the *MaxSp* can be determined by seeking a threshold that makes the cluster the "thinnest". As in Fig. 3, when *Eps* and *MinNum* are fixed, the core object number is changing with respect to the *MaxSp* value. The *MaxSp* threshold is determined with the highest gradient, which appears as the first jump point in Fig. 3. Here all objects with the spatial entropy value higher than the threshold (above the line) are noise and the others are core objects.

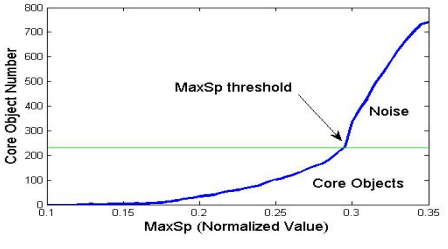


Fig. 3. Core objects number vs. *MaxSp* for the sample dataset in Fig. 1 (a)

4 Experiments

In this section, we evaluate SEClu by applying to both synthetic datasets and a real-world dataset. All experiments are performed on a 2.8GHz PC with 3G memory.

Experiment one – *spatial clustering of synthetic datasets*
This experiment shows the accuracy of SEClu for identifying clusters with spatial correlation. Fig. 4 shows three sample datasets. Each dataset includes x, y coordinates as the spatial attribute and a grey/intensity value as the non-spatial attribute.

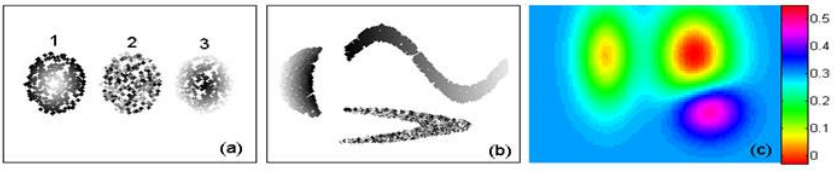


Fig. 4. Three synthetic spatial datasets. (c) is with color in electronic version.

The three datasets have different shapes and follow different spatial correlation functions. Fig. 4 (a) includes three round shaped groups. The grey values of points in group 1 decrease as an exponential function with distance from the center. The grey values of group 2 are random. The grey values in group 3 increase linearly with distance to the center. Since both groups 1 and 3 have strong spatial correlation they should be identified as clusters. Group 2 does not form a cluster with spatial correlations and should be labeled as noise. Fig. 4 (b) shows three irregularly shaped groups. The S-shaped and new moon shaped groups have clusters with spatial correlation while the objects in the V-shaped group have random non-spatial attributes and should be identified as noise. Fig. 4 (c) shows a raster dataset (i.e. the objects are distributed cell by cell) with a spatial correlated non-spatial attribute. Three clusters exist in the dataset with the non-spatial attribute values varying significantly at the boundary of each cluster.

For SEClu, we set the parameters ($MinNum$=18, Eps=0.017, $MaxSp$=0.29) for datasets (a) and (b) in Fig. 4. Since dataset (c) has evenly distributed spatial attributes, we apply a 5×5 window to decide the neighbor objects in $Eps(p_i)$, and $MaxSp$ is set as 0.3. Fig. 5 shows the clustering results on three datasets. From the figure, it is evident that: first, SEClu discovers clusters with spatial correlations successfully. In Fig. 5 (a), data groups with either high value center correlation or low value center correlation are identified as clusters, while the random one is labeled as noise. Second, since SEClu is a density-based clustering method, it can discover clusters with irregular shapes, e.g., SEClu identifies the S-shaped and the new moon shaped spatial correlated clusters. Third, SEClu finds clusters with both spatial correlation and high local non-spatial similarity. For example, in Fig. 5 (c) three spatial correlated clusters locate very close to each other. Measuring the local non-spatial attribute similarity, SEClu successfully separates the three clusters by detecting the significant dissimilarities in the non-spatial attribute.

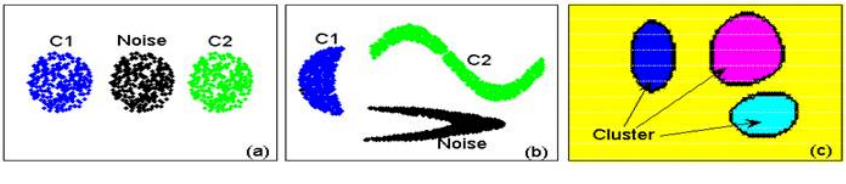


Fig. 5. SEClu clustering results. (The electronic version is with color).

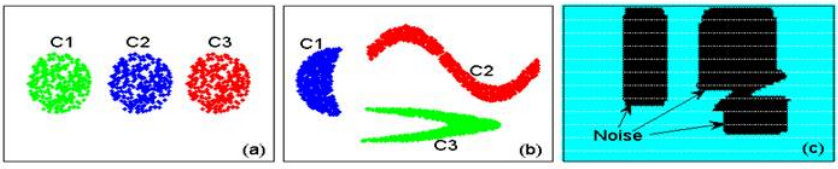


Fig. 6. GDBSCAN clustering results. (The electronic version is with color).

We also compare SEClu with GDBSCAN [1]. In GDBSCAN, the non-spatial attribute is treated similar to the spatial attributes and has a "weight". In the experiments, for datasets (a) and (b), the grey value is linearly normalized to the range of the x coordinate and can be viewed as the third coordinate. The parameters to GDBSCAN are set as ($MinPts$=18, Eps=0.017), which is similar to the SEClu configuration. For dataset (c), the intensity value is linearly normalized into [0,1] and the non-spatial attribute weight function keeps the intensity difference in a small range, i.e. $max|color(p_i) - color(p_j)| < 0.1$, where p_i and p_j are two arbitrary objects in the neighborhood. We also use a 5×5 window as the neighborhood.

Table 1. Accuracy comparision between SEClu and GDBSCAN

Dataset (No. of data objects)	SEClu			GDBSCAN		
	Correct	Error	Accuracy	Correct	Error	Accuracy
Dataset(a) (900 pts)	900	0	**100%**	600	300	**66.7%**
Dataset(b) (1500 pts)	1500	0	**100%**	1000	500	**66.7%**
Dataset(c) (6000 cells)	6000	0	**100%**	3963	2037	**66.1%**

Fig. 6 shows the clustering result of GDBSCAN. In datasets (a) and (b), considering the non-spatial attribute as independent from spatial attributes GDBSCAN incorrectly labels data with random distributed grey values as clusters. In dataset (c), even though spatial correlation exists in the three clusters, the non-spatial attribute changes. Since GDBSCAN only considers non-spatial attribute similarity, it is not suitable for identifying clusters in which the values of the non-spatial attribute are dissimilar but that still follow spatial correlations. Compared with the results of GDBSCAN, SEClu finds more meaningful clusters.

A detailed accuracy comparison between SEClu and GDBSCAN is shown in Table 1. From the table, the accuracy of SEClu to the three datasets is 100% while GDBSCAN is around 66%, which demonstrates SEClu performs better than GDBSCAN in identifying clusters with spatial correlations.

Experiment Two – *real application to the census map of Alberta*
The second experiment is performed on the 2006 Census dataset of Alberta, Canada. In this experiment, our goal is to identify clusters containing strong spatial patterns between population and community locations. There are 5181 community records available in the dataset. Each record is represented as a point, with the spatial attributes referring to the coordinates and the non-spatial attribute referring to the population.

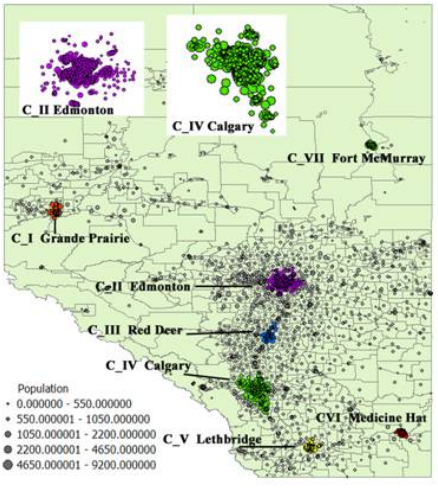


Fig. 7. Alberta census data clustering results from SEClu

We set (*Eps*=0.13, *MinNum*=15, *MaxSp*=0.3) to SEClu, and the clustering results are shown in Fig. 7 overlaid on the map of the Province of Alberta. In Fig. 7, clusters identified by SEClu, exhibit meaningful spatial patterns. All 7 clusters are around major cities in Alberta, which are downtown centered and have a radial shape extending to the suburb area. Also, the clustered communities close to city centers tend to have more population than those locating in the suburb, which reveals the phenomenon that many more people live in cities as compared to rural areas in Alberta. For example, the population in the cluster area reaches 2.56 million, which counts 76% of the total population in Alberta. This is consistent with records from Alberta Municipal Affairs that 81% of the Alberta population lives in urban cities and only 19% lives in rural area.

5 Conclusions

In this paper, we first demonstrate that spatial entropy is a decreasing function when spatial correlation and local non-spatial attribute similarity increase. Then, we propose a novel spatial clustering method, termed SEClu, that is based around the use of spatial entropy. SEClu considers spatial attributes, non-spatial attributes and spatial correlation during the clustering process and can identify arbitrary shaped clusters with spatial correlations. In our experiments, we apply SEClu to synthetic datasets and to a real-world application, and compare the clustering results with those of GDBSCAN. Results show that SEClu can discover meaningful spatial clusters, and perform better than GDBSCAN at finding clusters with spatial correlations.

In the future, we will improve SEClu in terms of the following aspects. First, we will further evaluate SEClu with larger dataset and extend it to be able to deal with more spatial data types, such as polygons. Second, we will apply SEClu to real-world geological datasets, which usually have large amounts of data, complicated spatial correlations and have significance for real applications.

References

1. Sander, J., Ester, M., Kriegel, H.P., Xu, X.W.: Density-based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications. Data Mining and Knowledge Discovery 2(2), 169–194 (1998)
2. Shekhar, S., Chawla, S.: Spatial Databases A Tour. Pearson Education, London (2003)
3. Wang, X., Hamilton, H.J.: DBRS: A Density-based Spatial Clustering Method with Random Sampling. In: Whang, K.-Y., Jeon, J., Shim, K., Srivastava, J. (eds.) PAKDD 2003. LNCS (LNAI), vol. 2637, pp. 563–575. Springer, Heidelberg (2003)
4. Claramunt, C.: A Spatial Form of Diversity. In: Cohn, A.G., Mark, D.M. (eds.) COSIT 2005. LNCS, vol. 3693, pp. 218–231. Springer, Heidelberg (2005)
5. Ng, R.T., Han, J.W.: CLARANS: A Method for Clustering Objects for Spatial Data Mining. IEEE Transactions on Knowledge and Data Engineering 14(5), 1003–1016 (2002)
6. Kailing, K., Kriegel, H.-P., Pryakhin, A., Schubert, M.: Clustering Multi-represented Objects with Noise. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 394–403. Springer, Heidelberg (2004)
7. Li, X., Claramunt, C.: A Spatial Entropy-Based Decision Tree for Classification of Geographical Information. Transactions in GIS 10(3), 451–467 (2006)
8. Tobler, W.R.: Computer Movie Simulating Urban Growth in Detroit Region. Economic Geography 46(2), 234–240 (1970)
9. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: 2nd International Knowledge Discovery and Data Mining, pp. 226–231. AAAI Press, Portland (1996)
10. Sturges, H.A.: The Choice of A Class Interval. Journal of the American Statistical Association 21, 65–66 (1926)

Self-adjust Local Connectivity Analysis for Spectral Clustering

Hui Wu[1], Guangzhi Qu[1], and Xingquan Zhu[2]

[1] Oakland University, Rochester MI 48309, USA
{hwu,gqu}@oakland.edu
[2] QCIS Centre, University of Technology, Sydney, NSW 2007, Australia
xqzhu@it.uts.edu.au

Abstract. Spectral clustering has been applied in various applications. But there still exist some important issues to be resolved, among which the two major ones are to (1) specify the scale parameter in calculating the similarity between data objects, and (2) select propoer eigenvectors to reduce data dimensionality. Though these topics have been studied extensively, the existing methods cannot work well in some complicated scenarios, which limits the wide deployment of the spectral clustering method. In this work, we revisit the above two problems and propose three contributions to the field: 1) a unified framework is designed to study the impact of the scale parameter on similarity between data objects. This framework can easily accommodate various state of art spectral clustering methods in determining the scale parameter; 2) a novel approach based on local connectivity analysis is proposed to specify the scale parameter; 3) propose a new method for eigenvector selection. Compared with existing techniques, the proposed approach has a rigorous theoretical basis and is efficient from practical perspective. Experimental results show the efficacy of our approach to clustering data of different scenarios.

Keywords: spectral clustering, scale parameter, eigenvector selection.

1 Introduction

Clustering is a common procedure of statistical data analysis, which is extensively used in machine learning, data mining and pattern recognition([7], [16], [6], [9]). The goal of clustering is to partition a data set into a number of groups with high intra-group relevance and low inter-group relevance. Among all the clustering techniques, *spectral clustering* is popular and has many fundamental advantages ([11], [1], [23], [3], [22], [12], [4]). Spectral clustering stems from a strong theoretical foundation ([2], [5]) and its performance often outperforms other traditional approaches. Hence, spectral clustering has been successfully applied in many areas including image segmentations ([19],[13], [15], [28], [27], [20], [21], [20], [14]), bioinformatics ([5], [17], [18]), social network([10], [26], [24]), and document clustering ([8]). Among all the previous works, Ng-Jordan-Weiss'

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 209–224, 2011.

framework (NJW for short) lays a remarkable foundation in emphasizing the importance of expressing the data appropriately [15]. Firstly, NJW utilizes a weighted graph to represent the raw data. Then it calculates the eigenvectors of this weighted graph's Laplacian matrix and selects the top k biggest eigenvectors w.r.t the eigenvalues for further processing. Finally, the selected eigenvectors are normalized and k-means clustering method is utilized to cluster the normalized eigenvectors data. Apparently, how to build the weighted graph and further the quality of the weighted graph are critical for the clustering performance.

In NJW's framework, a global scale parameter is used in establishing the weighted graph from the raw data. But this solution ignores the difference among data points in term of their local shapes. To fix this weakness, Zelnik-Manor and Perona (ZP) [28] specify a local scaling parameter for each data point. In their method, the distance between a point and its K-th neighbor is selected as this point's scale parameter. Though this approach takes into account a point's local shape characteristics to some extent, it may neglect the facts that the resulting neighbor data points may belong to different groups. In seeking a better solution to remedy NJW's weakness, we propose in this paper another approach to specifying the local scaling parameter based on data local connectivity information, with an aim of providing a scale parameter self-adjusting spectral clustering algorithm.

Besides specifying the scaling parameter for each data point, another important problem in spectral clustering is to select appropriate eigenvectors for further processing (e.g., clustering). Recently, several works have been proposed. Xiang and Gong proposed a probability-based method to select eigenvectors [25]. In their approach, the elements of each eigenvector follow either unimodal or multmodal distributions based on whether the eigenvector is relevant. Under this formulation, every eigenvector has an importance score, which is used to select informative eigenvectors. Another work by Zhao *et al*. [29] evaluates the importance of an eigenvector by its entropy. Different from the existing approaches, our method utilizes both eigenvalues and eigenvectors in choosing most appropriate eigenvectors. Moreover, the proposed method is simple to implement yet very efficient in practice.

The contribution of this work is three-fold. First, we build a unified framework to study the impact of the scale parameter on calculating similarity between data objects. This framework can easily accommodate various state of art methods spectral clustering methods for performance comparison study. Second, we design a new approach to specifying the scale parameter based on local connectivity analysis. Third, we present an effective method for eigenvector selection. Compared with previous works, our solution has sound theoretical basis and is efficient from practical perspective. The experimental results show the efficacy of our approach in handling the data clustering of different scenarios.

2 Methodology

This work is inspired by Ng *et al.*[15] that establishes the spectral clustering framework (NJW for abbreviation) and Zelnik-Manor and Perona's attempt on

improving NJW method [28]. To make this paper self-contained, a concise introduction of NJW algorithm is presented in the following.

For a given data set $S = \{s_1, s_2, \ldots, s_n\}(s_i \in \mathbb{R}^l$, where $l \in \mathbb{R}$ is the total dimensions of the data object attributes space; n is the number of data objects), the goal is to partition S into k different subsets. Algorithm 1 describes the NJW framework.

Algorithm 1. Ng-Jordan-Weiss Algorithm

1: Form the affinity matrix $A \in \mathbb{R}^{n \times n}$:

$$A_{ij} = \begin{cases} \exp(-d(s_i, s_j)^2/2\sigma^2) & i \neq j; \\ 0 & i = j. \end{cases} \quad (1)$$

where σ is a scale parameter, and $d(s_i, s_j)$ is a distance function, such as Euclidean distance formula;

2: Construct the Laplacian matrix $L = D^{-1/2}AD^{-1/2}$ of the weighted graph, where D is a diagonal matrix, $D_{ii} = \sum_j A_{ij}$;

3: Get an matrix $X = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{n \times k}$, where x_i is an eigenvector and it corresponds to the i'th largest eigenvalue of L;

4: Normalize matrix X into matrix Y: $Y_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$

5: Clustering rows in Y with k-means.

As shown in Algorithm 1, there are two data representation transformations. The first one is that the raw data S is represented by the affinity matrix A. The second is to represent the data by top K eigenvectors of the Laplacian matrix, denoted by X. In this paper, we focus on how to improve the spectral clustering from each transformation. In Section 2.1, we propose a local connectivity-based method to specify the scale parameter, which produces a high quality affinity matrix proved by the experimental evaluation. In Section 2.2, method taking advantage of both eigenvectors and eigenvalues is proposed to choose eigenvectors.

2.1 Local Connectivity-Based Scaling

While representing the raw data with the affinity matrix(which is also named as the weighted graph) an ideal transformation of the data is to allocate the data points in a same group closely in the new representation space, and data points from different groups far away from each other. Therefore, different groups have weak connectivities, which is helpful to separate the data objects correctly ([2], [5]). Let us give one example to explain the impact of different affinity matrices on representing the data.

Figure 1(a) is the desired clustering result, where there are two groups of data points: a fish shape group and a line shaped group. In Figures 1(b) to 1(d), we use the horizontal axis and the vertical axis to represent the first and second eigenvectors, respectively. By comparing these three figures, we can easily draw the conclusion that the representation in Figure 1(b) is better for clustering the data, since points in different groups are separated clearly. On the other hand,

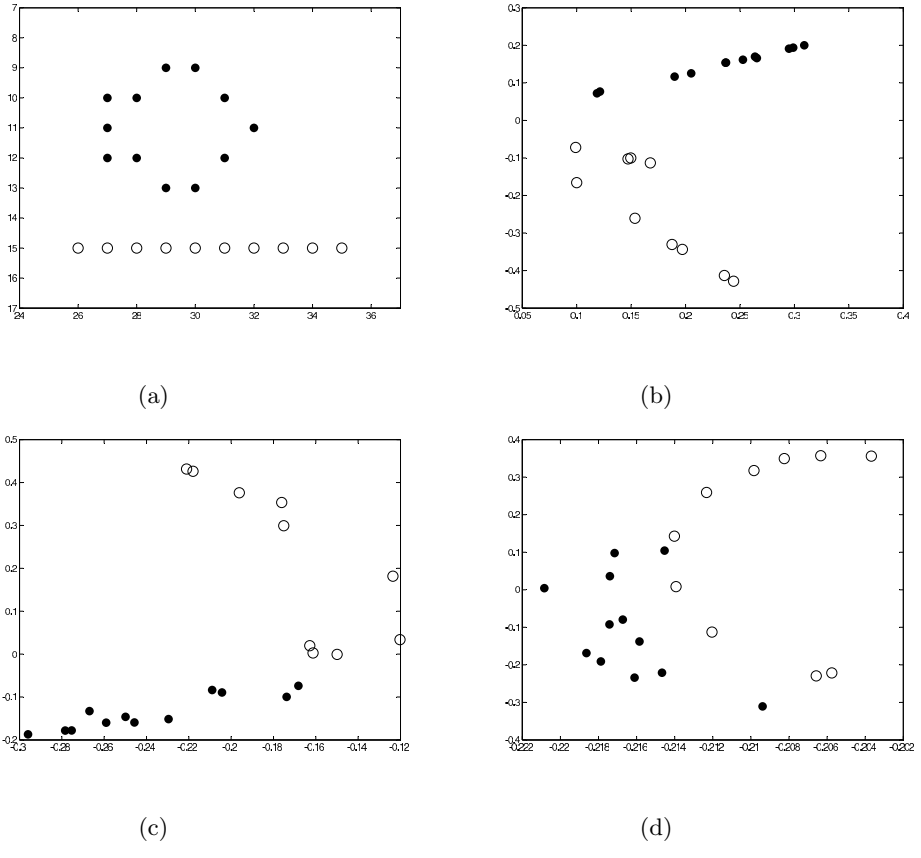


Fig. 1. Original Data Set and Examples of Different Representations with Top 2 Eigenvectors of Affinity Laplacian Matrix L

shadow and hollow points are quite mixed each other in Figures 1(c) and 1(d), which shows the difficulty in correct clustering. Actually, these three affinity matrices are generated by three different parameter values, which will be introduced in more details later. In spectral clustering, all affinities are calculated by Equation 1. In this paper, we focus on how to improve this calculation (Equation 1), and further the quality of the affinity matrix.

Before giving a detailed description of the proposed approach, we firstly introduce a concept of *Core Set*. Consider a point $s_i \in S$, its *Core Set*, denoted as $C(s_i)$, includes all the data points that have high affinities to s_i. Let α be the threshold to determine whether there exists an affinity or not. $C(s_i)$ is defined as:

$$C(s_i) = \{s_j | A_{ij} \geq \alpha, s_j \in S\}. \tag{2}$$

Let σ_{s_i} be the specified scale parameter for s_i, then:

$$\alpha = \exp(\frac{-1 \times R^2(s_i)}{2 \times \sigma_{s_i}^2}). \tag{3}$$

where $R(s_i)$ is the *Core Radius* of $C(s_i)$, which is defined by:

$$R(s_i) = \max_{s_j \in C(s_i)} d(s_j, s_i) \tag{4}$$

Therefore, if point s_j has the distance of $R(s_i)$ to point s_i, the affinity between s_i and s_j will be quantified as α. Given α and σ_{s_i}, $R(s_i)$ can also be calculated by:

$$R(s_i) = \sqrt{-2 \times \ln(\alpha) \times \sigma_{s_i}^2}. \tag{5}$$

In NJW algorithm, σ is a user-customized global parameter, which means for each data point, its σ_{s_i} is always equal to the assigned σ. Furthermore, α, used to define a small similarity value, is also a global parameter. According to Equation 5, the *Core Radius* values of all points are equal. Unfortunately, this global scaling scheme can easily cause two unexpected situations, namely, *partial-fitting* and *over-fitting*, which are illustrated in Figures 2(a) and 2(b).

Figure 2(a) shows the scenario of *partial-fitting*. For a point s_i, *partial-fitting* means that there exists at least one point whose desired class is the same as s_i, but it is not in $C(s_i)$. Let $L(s_i)$ be a function to denote s_i's true clustering label, *partial-fitting* can be formulated as:

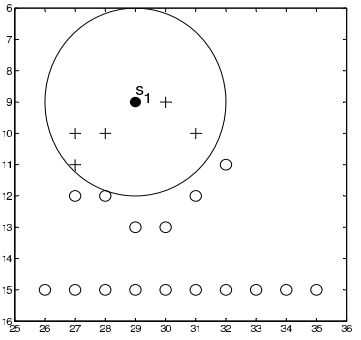
$$|\{s_j | s_j \notin C(s_i) \; and \; L(s_i) = L(s_j), s_j \in S\}| \geq 1 \tag{6}$$

Figure 2(b) shows the scenario of *over-fitting*. *over-fitting* means there exists at least one such point, whose desired class is not the same as s_i's, but it is involved in s_i's *Core Set*. Formally,

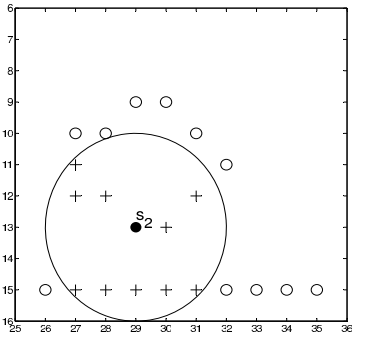
$$|\{s_j | s_j \in C(s_i) \; and \; L(s_i) <> L(s_j), s_j \in S\}| \geq 1 \tag{7}$$

To fix this weakness, Zelnik-Manor and Perona utilized a point's neighboring information to specify the scale parameter. In their method, σ_{s_i} is equal to the distance between point s_i and its K-th closest neighbor. Therefore according to Equation 5, the *Core Radius* value will be different for different points. This method improves NJW spectral clustering to certain extent. However, it cannot resolve the problem completely. Figures 2(c) and 2(d) show the results of their method, where K is equal to 4. Apparently, *partial-fitting* and *over-fitting* problems do exist.

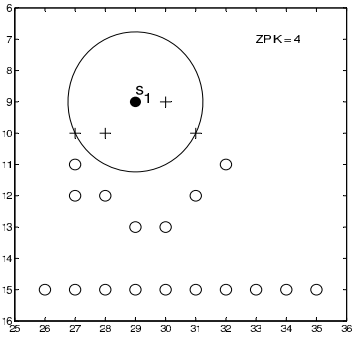
The essential reason that both NJW and ZP algorithms cannot overcome *partial-fitting* and *over-fitting* problems is that their approaches cannot guarantee that all the points in $C(s)$ have the same desired cluster label as s's. In order to solve these problems, we propose an approach based on local connectivity analysis that the data point can utilize its local information to self adjust the scale parameter. Figures 2(e) and 2(f) show the results from our method for data



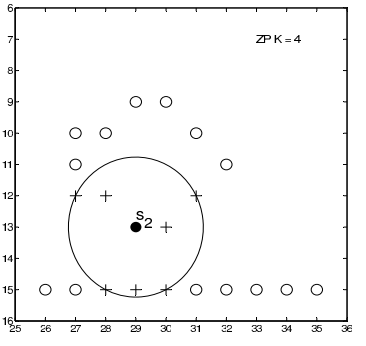
(a) $C(s_1)$ in NJW Algorithm



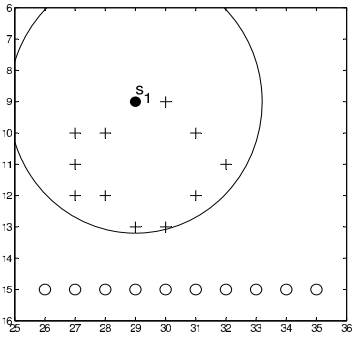
(b) $C(s_2)$ in NJW Algorithm



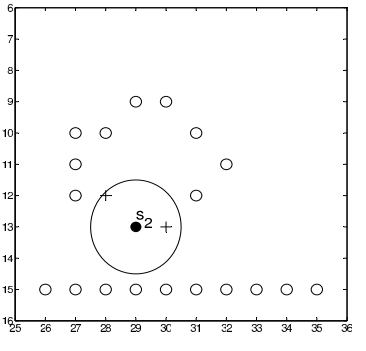
(c) $C(s_1)$ in ZP Algorithm



(d) $C(s_2)$ in ZP Algorithm



(e) $C(s_1)$ in Our Algorithm



(f) $C(s_2)$ in Our Algorithm

Fig. 2. Instances of *Core Set* in Different Algorithms. The cycle indicates the *Core Set*, where its center is the currently considered point s_i and its radius is equal to s_i's *Core Radius*.

points s_1 and s_2. Obviously, all points in $C(s_1)(C(s_2))$ share the same expected cluster label of $s_1(s_2)$ so there is no *over-fitting*. Our approach avoids the *partial-fitting* problem for data point s_1 and *over-fitting* problem for data point s_2, which were suffered from other approaches shown in Figures 2(a) through 2(d).

In the following, we introduce the concept of *adjacent region* for a data point s in a multi-dimension space. It is denoted by $\eta(s)$ and formally defined as:

$$\eta(s) = \{p | s^i - r \leq p^i \leq s^i + r, 1 \leq i \leq m\} \tag{8}$$

where s^i denotes the i-th dimension of data point s, and r is a user specified constant parameter. In this work, we set r is equal to 1. It is easy to check that if $s_i \in \eta(s_j)$, then $s_j \in \eta(s_i)$. Either $s_j \in \eta(s_i)$ or $s_i \in \eta(s_j)$ implies that s_i, s_j are *adjacent*.

Let $s_i, s_j \in S$ be two data points, s_i, s_j are said to be *connected*, if (1) $s_i \in \eta(s_j)$, or (2) there exists a point sequence $s_i, s_{t_1}, \ldots, s_{t_m}, s_j$, where $s_{t_1} \in \eta(s_i)$, $s_{t_p} \in \eta(s_{t_{p+1}})$ for $1 \leq p \leq m-1$ and $s_{t_m} \in \eta(s_j)$; otherwise, s_i, s_j are *disconnected*. For simplicity purpose, we define this point sequence as a *path*, denoted as p_{s_i,s_j}. If all points in $p(s_i, s_j)$ are in a point set U, we call that p_{s_i,s_j} is *saturated* in U, denoted as $\chi(p_{s_i,s_j}, U)$. With the above notations, we can redefine *Core Set* of s_i as:

$$\begin{aligned} &C(s_i) = \arg\max_{U \subseteq S} |U| \\ &s.t. \quad \forall s_j \in U, \exists p_{s_i,s_j} \; and \; \chi(p_{s_i,s_j}, U). \end{aligned} \tag{9}$$

As shown in Equation 9, we employed two principles in forming the *Core Set*: no *over-fitting* and minimize *partial-fitting*. With the constraint, we can avoid *over-fitting* absolutely. The objective function tries to maximize the size of the core set, which is equivalent to minimizing *partial-fitting* problem. Figure 3 illustrate the concept with an example.

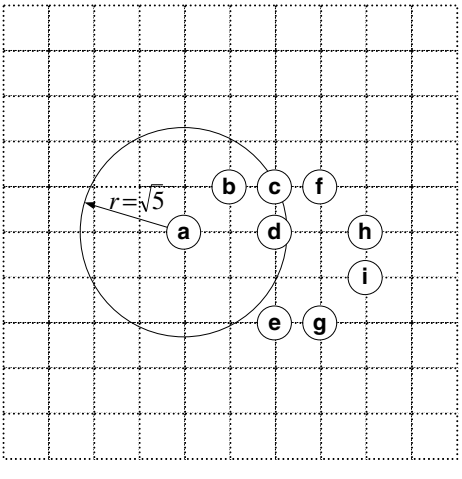


Fig. 3. Core Set and Core Radius Illustration

There are nine points labeled $a, b, \ldots, i$ in Figure 3. All the points are scattered in a grid space where each grid cell has a length of one unit. When consider the point a, using the principle of *no over-fitting*, we can expand and find its *Core Set* according to the formulation in Equation 9. According to the criterion defined in Equation 8 in determining adjacency relationship, points b, c, d are supposed to be involved in point a's *Core Set*. The distance between points a and c is $\sqrt{5}$. If we further expand the *Core Set*, the next point being considered should be e. However, the path a, b, c, f, h, i, g, e is not *saturated* in the set of $\{a, b, c, d, e\}$. Therefore, e can not be involved. So point a's *Core Set* is $\{b, c, d\}$, and its *Core Radius* $R(a)$ is equal to $\sqrt{5}$ according to Equation 4.

Consider two points s_i and s_j, the affinity between them is different when we look it from distinct viewpoints. From point s_i, the affinity is $\exp(-1 \times d^2(s_i, s_j)/(2 \times \sigma_i^2)$, while it is $\exp(-1 \times d^2(s_i, s_j)/(2 \times \sigma_j^2)$ from point s_j's perspective. We unify the affinity between points s_i and s_j by Equation 10.

$$A_{ij} = \max(\exp(\frac{-d^2(s_i, s_j)}{2 \times \sigma_i^2}), \exp(\frac{-d^2(s_i, s_j)}{2 \times \sigma_j^2})) \tag{10}$$

In addition, for the point itself, its own affinity A_{ii} is defined to be equal to 0.

In our approach, parameter α needs to be specified. We will show in the following that this is an easy task. Clustering performance can be evaluated by the difference between the intra-cluster affinity summation and the inter-cluster affinity summation, which is given as:

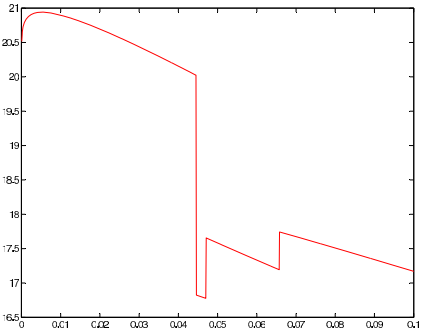
$$\phi(V, A) = \max_V \frac{1}{2}(\sum_{t=1}^{k} \sum_{i,j \in V_t} A_{ij} - \sum_{t=1}^{k} \sum_{i \in V_t, j \notin V_t} A_{ij}) \tag{11}$$

where A is a given data set, k is the total number of clusters, V is the clustering result, and V_t indicates the point set of the t^{th} cluster. The ideal performance ϕ^* of clustering on A is defined as:

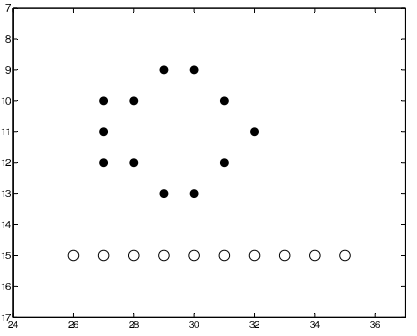
$$\phi^* = \max_V \phi(V, A) \tag{12}$$

We utilize an example to illustrate the relation between α and ϕ^* in Figure 4. In Figure 4(a), the horizontal axis represents different α values, and the vertical axis shows the clustering performance values of ϕ^*. As depicted in Figure 4(a), there is a high ϕ^* area, which corresponds to a value range of α close to 0. When α is equal to 0.0052, ϕ^* achieves the highest value of 20.9393, and the data set will be clustered correctly as shown in Figure 4(b). When α value increases, ϕ^* value will decrease (clustering performance drops). When α value is too big, it is difficult to cluster data set correctly. This scenario is reflected in Figures 4(c) and 4(d), when α is equal to 0.045 and 0.1, respectively.

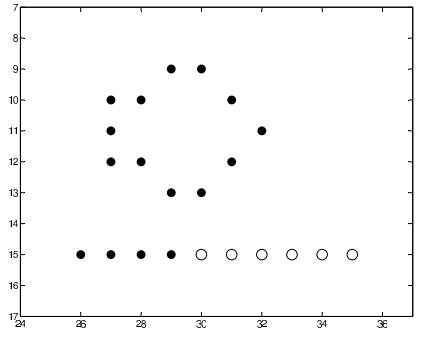
Theoretically, α is supposed to be a small value. Recall the process of forming the *Core Set*, we prefer the data points involved to have the same clustering results as the one being considered. According to Equation 11, a good clustering result should have big difference between the internal affinity summation and the



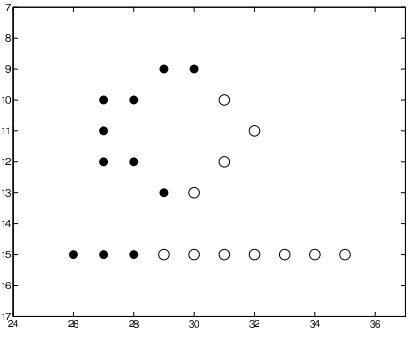
(a) a function between α and ϕ^*



(b) clustering result when $\alpha = 0.0052$



(c) clustering result when $\alpha = 0.045$



(d) clustering result when $\alpha = 0.1$

Fig. 4. Clustering Results with Different α values On Example Data Set

external affinity summation. Therefore, a simple way is to let points outside of s_i's *Core Set* have zero affinity to s_i. In other words, the points whose distances to s_i are equal to the *Core Radius* should have a small affinity. On the other hand, if let s_i have high affinity to all other points, ϕ^* will be small, which has been proved by Figure 4(a). Practically, α can be regarded as a small value constant, say *0.0001*.

2.2 Eigenvector Selection

Recently, eigenvector selection has been deployed to improve the performance of spectral clustering [25] [29]. In this work, we proposed a simple yet effective strategy, which considers both eigenvalues and eigenvectors, to select eigenvectors for clustering. Eigenvalues are used to determine the candidate eigenvectors. Further decisions are based on analyzing the selected eigenvectors.

The goal of eigenvector selection is to select informative eigenvectors as the representation of the original data objects set. Different from the method given in NJW framework (where the top K biggest eigenvectors are selected), we firstly

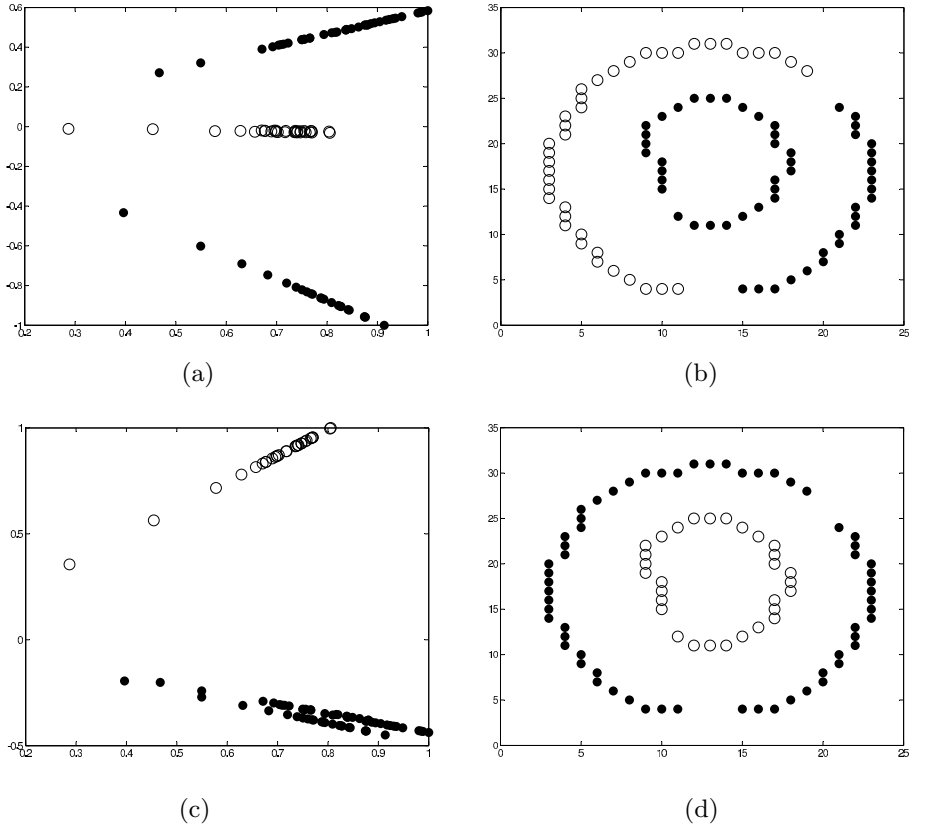


Fig. 5. Comparisons between Using and Not Using Eigenvector Selection

choose $K + t$ (t is a constant, and it is equal to 2 in this work) biggest ones as candidates. And then we employ the following criterion for further selection:

$$\rho(z) = \frac{var(z)}{mean(z)} = \frac{\sqrt{\sum_{i=1}^{n}(z_i - mean(z))}}{mean(z)} \tag{13}$$

Where z is an eigenvector in X, z_i is the i-th item in z. $mean(z)$ is the mean value of z, which is equal to $\frac{1}{n}\sum_{i=1}^{n} z_i$ and $var(z)$ is the standard deviation of its elements. We give one example to illustrate the effectiveness of this approach.

In Figure 5, there is a comparison between using and not using eigenvector selection. Figure 5(a) is drawn by top 2 eigenvectors without eigenvector selection, and Figure 5(b) is the corresponding clustering result of this approach. According to these two figures, we can learn that the top 2 eigenvectors are not always good candidates for clustering. Figure 5(c) is generated by the two eigenvectors selected through Equation 13, and Figure 5(d) shows its clustering result. Obviously, these two eigenvectors represent the data in a better way, and the clustering result is desirable.

3 Experimental Evaluation

To evaluate the performance of our approach, we compare it with both NJW and ZP methods. As there is a user-specified parameter in each algorithm, our comparison will focus on both the correctness of the results and the convenience of assigning the parameters.

On evaluating the convenience of assigning the parameter, we developed two metrics: *accuracy* and *coverage*. Let $D = \{d_1, d_2, \ldots, d_n\}$ $(n \in \mathbb{R})$ be a data set, p be a parameter and its value will be drawn from $v = \{v_1, v_2, \ldots, v_m\}(m \in \mathbb{R})$. Assume $f(v_i, d_j)$ be the function to judge whether an algorithm can cluster d_j correctly when $p = v_i$. If the algorithm can produce the desired result, then $f(v_i, d_j)$ is equal to 1; otherwise, we assign 0 to it. With these notations, *accuracy* is defined as:

$$accu(p = v_i, D) = \frac{\sum_{d_j \in D} f(v_i, d_j)}{|D|} \tag{14}$$

As shown in Equation 14, if $accu(p = v_i, D)$ is larger then more data points can be clustered correctly given $p = v_i$. In other words, it is easy for us to decide the value of parameter p. An ideal situation is that $accu(p = v_i, D)$ is equal to 1, then p can be treated as a constant. Metric *accuracy* is used to measure the effectiveness of clustering on entire training data set with a specified parameter value.

Metric *coverage* is introduced to quantify the complexity of determining the parameter to one sample, we define it as:

$$cover(p, d_j) = \frac{\sum_{v_i \in v} f(v_i, d_j)}{|v|} \tag{15}$$

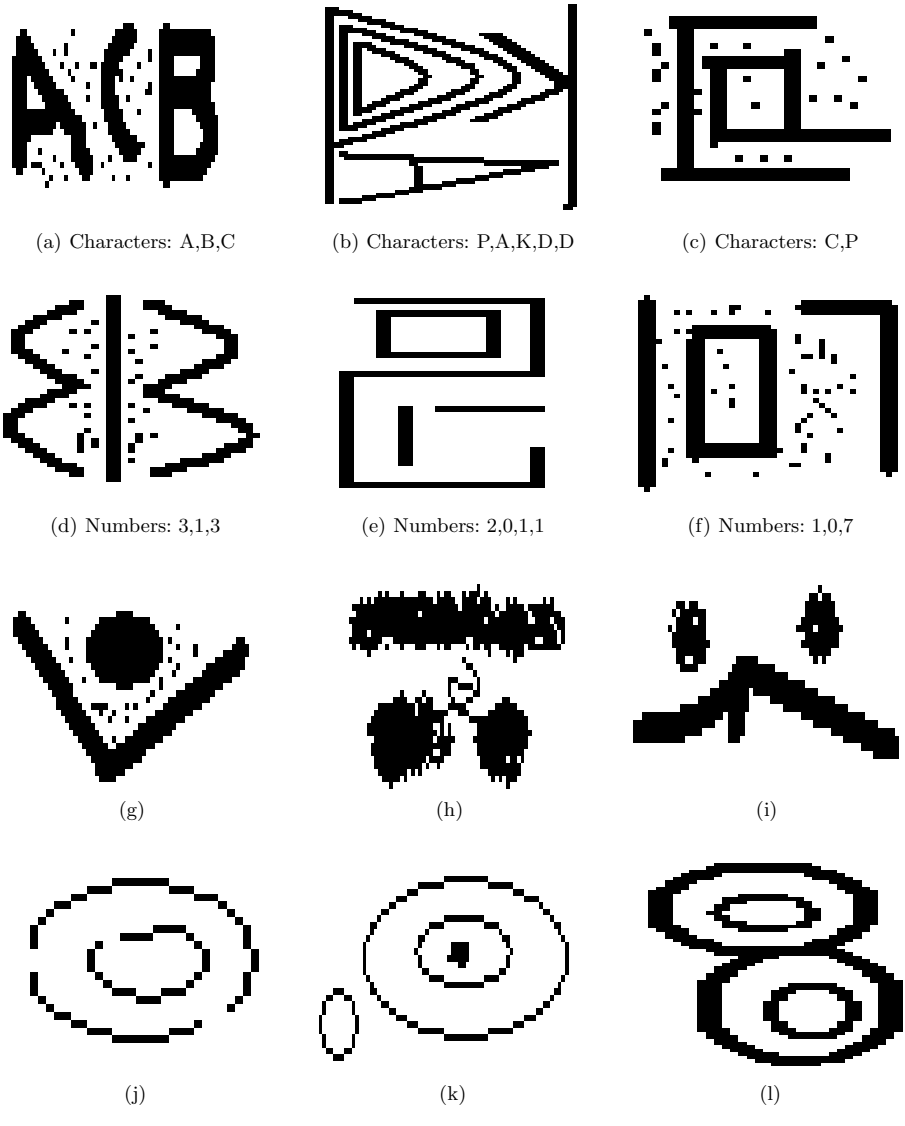
According to Equation 15, if $cover(p, d_j)$ is larger, it is easier to find a proper p value for d_j. Scenario $cover(p, d_j) = 1$ means any candidate parameter value can give the expected result.

To examine the effectiveness of the proposed approach, we evaluated 12 data sets in the experiments, which are shown in Figure 6. The data sets evaluated are roughly in four groups. Figures 6(a) to 6(c) are **Letters**. Figures 6(d) to 6(f) are **Numbers**. Figures 6(g) to 6(i) are **Dot-Line** in different shapes. Figures 6(j) to 6(l) are different combinations of **Cycles**.

These four classes of figures are very typical on analyzing the performance of spectral clustering, and similar data sets have been studied widely ([15], [28], [29]). The challenge in clustering **Letters** is that some letters are complex, such as 'A' and 'B', and if some letters are embedded into others, such as 'PAKDD' and 'CP', which makes the problem more complicated. Without proper scaling, most of the example figures cannot be clustered correctly. Figure 7(a) is K-means's result on the data in Figure 6(b).

Numbers is also a difficult task for clustering in many cases. For example, there are four numbers in Figure 6(e), and these numbers are mixed together. K-means cannot separate them correctly, whose result is shown in Figure 7(b).

The difficulty of clustering **Dot-Line** data sets is that the segment of the *line* which is very close to a dot can be easily clustered into the *dot* group. Figure 7(c) gives one example of this scenario.



(a) Characters: A,B,C (b) Characters: P,A,K,D,D (c) Characters: C,P

(d) Numbers: 3,1,3 (e) Numbers: 2,0,1,1 (f) Numbers: 1,0,7

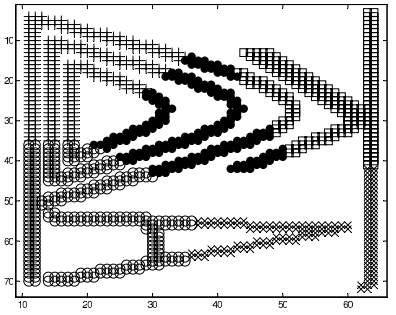
(g) (h) (i)

(j) (k) (l)

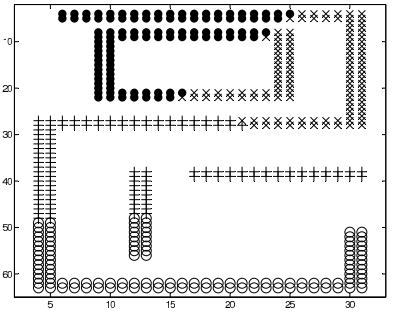
Fig. 6. Date Sets for Experiments

For **Cycles** data sets, more cycles may share the same center, then they are difficult to be clustered without re-presentation. Figure 7(d) is one example of this case.

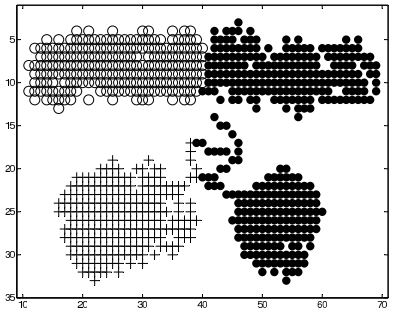
We test all three algorithms (NJW, ZP and ours) on all figures given in Figure 6. For each algorithm, we test its parameter within a range. For NJW algorithm, σ changes from 0.1 to 2.0, and its step of change is set to be 0.1. For k in ZP algorithm,



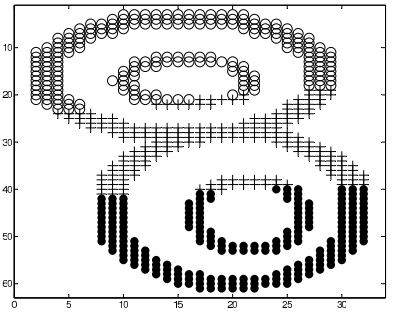
(a) K-means's Result to Figure 6(b)



(b) K-means's Result to Figure 6(e)

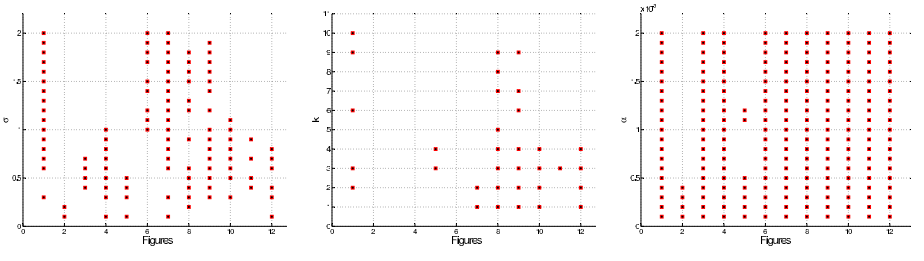


(c) K-means's Result to Figure 6(h)



(d) K-means's Result to Figure 6(l)

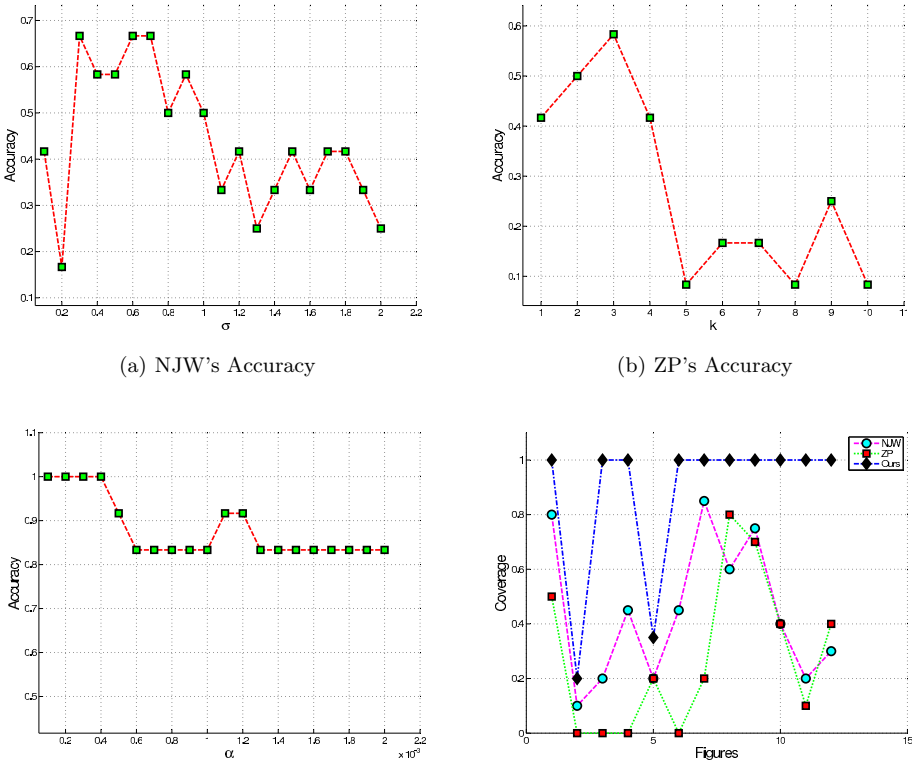
Fig. 7. Clustering Examples by Our Algorithm



(a) NJW Algorithm (b) ZP Algorithm (c) Our Algorithm

Fig. 8. Experimental Results of Three Algorithms

it ranges from 1 to 10, and its step length is 1. α in our algorithm is between 0.0001 to 0.02, and its step length is 0.0001. All the results of these algorithm are given in Figure 8. In these three figures, the horizontal axis indicates the data set indexes, where 1 to 16 correspond Figures 6(a) to 7(d), and the vertical axis represent the



(a) NJW's Accuracy (b) ZP's Accuracy

(c) Our Accuracy (d) Coverage Comparison

Fig. 9. Performance Comparison Results

parameter values of the algorithm. Furthermore, one square indicates a correct case. For example, there is a square at (1, 0.1) in Figure 8(c), which means that when $\alpha = 0.1$, Figure 6(a) can be clustered correctly.

The comparison results of *coverage* and *accuracy* on these three algorithms are given in Figure 9. As described in Figures 9(a) to 9(c), our algorithm works very well. When α is in $\{0.0001, 0.0002, 0.0003, 0.0004\}$, it reaches the ideal scenario - the *accuracy* is equal to 1. Therefore, we can simply take α as a constant, whose value is 0.0001. Furthermore α's accuracy is much higher than σ's and k's, and it also fluctuates much smaller than σ, k.

On the metric *coverage*, our parameter α also gets high performance, which is described in Figure 9(d). Surprisingly, α gets the coverage of 1 in 83.3% data sets (10/12). Due to the page limitation, we won't be able to show more scenarios that our algorithm can work effectively with $\alpha = 0.0001$.

It should be noticed that though connectivity plays an important role in our algorithm, we still can handle 'unexpected' situations. In Figure 6(j), while the outer cycle is broken, α's coverage of this data set is even 1.

4 Conclusions

In this paper, we proposed a scale parameter specification approach to improve the NJW algorithm [15] as a scale parameter self-adjusting one. Compared with the specification method given in [28], our approach is less sensitive to the datasets. We also presented a new method for eigenvector selection, which is easier to apply than current approaches [25], [29]. Experimental results and comparisons demonstrated that our algorithm can deal with different scenarios, and can handle more instances which ZP's algorithm cannot cluster well. In the future, we will test and improve the proposed approach through more real applications.

Acknowledgment

This work is partially supported by OU-Beaumont Multidisciplinary Research Award. Xingquan Zhu is sponsored by ARC Future Fellowship under Grant No. FT100100971.

References

1. Bach, F.R., Jordan, M.I.: Learning spectral clustering. In: Advances in Neural Information Processing Systems, vol. 16. MIT Press, Cambridge (2003)
2. Chung, F.R.K.: Spectral Graph Theory. CBMS Regional Conference Series in Mathematics, vol. 92. American Mathematical Society, Providence (February 1997)
3. Dhillon, I.S., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: KDD 2004: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 551–556. ACM, New York (2004)
4. Filippone, M., Camastra, F., Masulli, F., Rovetta, S.: A survey of kernel and spectral methods for clustering. Pattern Recogn. 41(1), 176–190 (2008)
5. Higham, D.J., Kalna, G., Kibble, M.: Spectral clustering and its use in bioinformatics. J. Comput. Appl. Math. 204(1), 25–37 (2007)
6. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. 31(3), 264–323 (1999)
7. Jain, A.K.: Data Clustering: 50 Years Beyond K-means. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 3–4. Springer, Heidelberg (2008)
8. Kamvar, S.D., Klein, D., Manning, C.D.: Spectral learning. In: IJCAI 2003, pp. 561–566 (2003)
9. Kleinberg, J.: An impossibility theorem for clustering. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Neural Information Processing Systems, vol. 14, pp. 446–453. MIT Press, Cambridge (2002)
10. Kurucz, M., Benczúr, A.A., Csalogány, K., Lukács, L.: Spectral clustering in social networks, pp. 1–20 (2009)
11. Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)

12. Luxburg, U.V., Bousquet, O., Belkin, M.: Limits of spectral clustering. In: Advances in Neural Information Processing Systems. MIT Press, Cambridge (2005)
13. Meila, M., Shi, J.: A random walks view of spectral segmentation (2001)
14. Meila, M., Shi, J.: Learning segmentation by random walks. In: NIPS, pp. 873–879 (2000)
15. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems, vol. 14 (2001)
16. Pavel, B.: Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA (2002), http://www.accrue.com/products/rp_cluster_review.pdf
17. Pentney, W., Meila, M.: Spectral clustering of biological sequence data. In: AAAI 2005: Proceedings of the 20th National Conference on Artificial Intelligence, pp. 845–850. AAAI Press, Menlo Park (2005)
18. Qin, G., Gao, L.: Spectral clustering for detecting protein complexes in protein-protein interaction (ppi) networks. Mathematical and Computer Modelling 52(11-12), 2066–2074 (2010); The BIC-TA 2009 Special Issue, International Conference on Bio-Inspired Computing: Theory and Applications
19. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2000)
20. Tolliver, D.A., Miller, G.L.: Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In: CVPR 2006: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1053–1060. IEEE Computer Society Press, Washington, DC, USA (2006)
21. Tung, F., Wong, A., Clausi, D.A.: Enabling scalable spectral clustering for image segmentation. Pattern Recogn. 43(12), 4069–4076 (2010)
22. Verma, D., Meila, M.: A comparison of spectral clustering algorithms. Technical report, Department of CSE University of Washington Seattle, WA 98195-2350 (2005)
23. von Luxburg, U., Belkin, M., Bousquet, O.: Consistency of spectral clustering (April 2008)
24. White, S., Smyth, P.: A spectral clustering approach to finding communities in graphs (2005)
25. Xiang, T., Gong, S.: Spectral clustering with eigenvector selection. Pattern Recogn. 41(3), 1012–1029 (2008)
26. Xu, K.S., Kliger, M., Chen, Y., Woolf, P.J., Hero III, A.O.: Revealing social networks of spammers through spectral clustering. In: ICC 2009: Proceedings of the IEEE International Conference on Communications, Piscataway, NJ, USA, 2009, pp. 735–740. IEEE Press, Los Alamitos (2009)
27. Yu, S.X., Shi, J.: Multiclass spectral clustering. In: ICCV 2003: Proceedings of the Ninth IEEE International Conference on Computer Vision, p. 313. IEEE Computer Society, Washington, DC, USA (2003)
28. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: Advances in Neural Information Processing Systems (2004)
29. Zhao, F., Jiao, L., Liu, H., Gao, X., Gong, M.: Spectral clustering with eigenvector selection based on entropy ranking. Neurocomput. 73(10-12), 1704–1717 (2010)

An Effective Density-Based Hierarchical Clustering Technique to Identify Coherent Patterns from Gene Expression Data

Sauravjyoti Sarmah, Rosy Das Sarmah, and Dhruba Kumar Bhattacharyya

Dept. of Comp Sc & Engg., Tezpur University *, Napaam 784 028, India
sjs@tezu.ernet.in, rosy8@tezu.ernet.in, dkb@tezu.ernet.in

Abstract. We present an effective tree-based clustering technique (*Gene ClusTree*) for finding clusters over gene expression data. *GeneClusTree* attempts to find all the clusters over subspaces using a tree-based density approach by scanning the whole database in minimum possible scans and is free from the restrictions of using a normal proximity measure [1]. Effectiveness of *GeneClusTree* is established in terms of well known z-score measure and p-value over several real-life datasets. The p-value analysis shows that our technique is capable in detecting biologically relevant clusters from gene expression data.

Keywords: Maximal space cluster, reduced space cluster, coherent patterns, z-score, p-value.

1 Introduction

According to [2], most data mining algorithms developed for microarray gene expression data deal with the problem of clustering. Clustering groups of genes with similar expression patterns into the same cluster based on a proximity measure. One of the characteristics of gene expression data is that it is meaningful to cluster both in terms of genes or samples. Co-expressed genes can be grouped into clusters based on their expression patterns ([3], [4]). In *gene-based* clustering, the genes are treated as the objects, while the samples are the features. In *sample-based* clustering, the samples can be partitioned into homogeneous groups where the genes are regarded as features and the samples as objects. Both the gene-based and sample-based clustering approaches search exclusive and exhaustive partitions of objects that share the same feature space. Apart from these, a third category, that is *subspace clustering*, captures clusters formed by a subset of genes across a subset of samples. For subspace clustering algorithms, either genes or samples can be regarded as objects or features. The goal of gene expression clustering is to group co-expressed genes together. Co-expressed genes indicate co-function and co-regulation [5]. The purpose of clustering gene expression data is to reveal the natural structure inherent in the data. A good clustering

* The department is funded by DRS Phase-I under SAP of UGC.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 225–236, 2011.

algorithm should depend as little as possible on prior knowledge, for example requiring the pre-determined number of clusters as an input parameter. Clustering algorithms for gene expression data should be capable of extracting useful information from noisy data. Gene expression data are often highly connected and may have intersecting and embedded patterns [6]. Therefore, algorithms for gene-based clustering should be able to handle this situation effectively. Finally, biologists are not only interested in the clusters of genes, but also in the relationships (*i.e.*, closeness) among the clusters and their sub-clusters, and the relationship among the genes within a cluster (*e.g.*, which gene can be considered as the representative of the cluster and which genes are at the boundary area of the cluster). A clustering algorithm, which also provides some graphical representation of the cluster structure is much favored by the biologists and therefore a tree structured clustering of gene expression data helps biologists to identify the inherent details. A large number of clustering techniques have been been reported for analyzing gene expression data, such as Partitional clustering such as K-means [7], Fuzzy c-means [8], Quality Threshold Clustering [9] and Self Organizing Maps (SOMs) [10], Hierarchical clustering (Unweighted Pair Group Method with Arithmetic Mean, UPGMA) [4], Self-Organizing Tree Algorithm [11], Divisive Correlation Clustering Algorithm (DCCA) [12] and Dynamically Growing Self-Organizing Tree (DGSOT) [13], Density-based clustering such as Kernel Density-based [14], Density-based Hierarchical Clustering (DHC) [6], Regulation based density clustering (RDClust) [15], a recent density gene clustering (DGC) [16] and Shared nearest neighbor based clustering [17], Model-based methods like Self-Organizing Maps [10], neural networks [18], Graph-theoretic clustering (Cluster Affinity Search Techniques (CAST) [3], CLuster Identification via Connectivity Kernels (CLICK) [19], E-CAST [20] and graph clustering algorithm [21]. A recent graph based automated clustering technique (GCE) is presented in [22]. In [23], a two stage clustering algorithm for gene expression data using genetic algorithms is presented. A novel multi-objective genetic fuzzy clustering followed by support vector machine based classification is presented in [24]. Recently, a subspace clustering algorithm, CLIC, has been proposed in [25]. Majority of the clustering techniques are dependent on a choice of proximity measure. Also, due to the inherent high dimensionality and presence of noise in gene expression data, it is a challenging task to find the clusters inherent in the subspaces of the dataset. The hierarchical approach of clustering genes helps in visualizing the clusters at different levels of hierarchy. Also, it is important to establish that the clusters obtained are biologically relevant. In this paper, we introduce an effective clustering technique (*GeneClusTree*), which attempts to find all the possible clusters over subspaces and represents the results in a tree based structure. The proposed *GeneClusTree* can be found significant in view of the following basic advantages: (i) capable of identifying biologically relevant clusters of all shapes in presence of noise, (ii) does not require the number of clusters apriori, and (iii) free from the restrictions of using any specific proximity measure.

2 *GeneClusTree*

GeneClusTree is a gene based clustering technique which attempts to cluster the gene dataset using a tree-based density approach. *GeneClusTree* groups together genes with similar expression patterns and the tree structure provides a natural way to graphically represent the data set. At first, the gene expression data is normalized to have mean 0 and standard deviation 1. Expression data having a low variance across conditions as well as data having more than 3-fold variation are filtered out. Discretization is then performed on this normalized expression data by retaining the up- or down- regulation information in each of the conditions for a particular gene. Also, to avoid the restrictions caused due to the use of any normal proximity measure, it exploits the angle information computed over the normalized expression values.

2.1 Regulation Information Extraction

Let G be the set of all genes, T be the set of all conditions with cardinality n_T. The regulation information extraction for a gene $g_i \in G$ is carried out as follows:

(i) ξ of gene g_i at the first condition, *i.e.*, at t_1 is:	(ii) Similarly, ξ of gene g_i at conditions t_j $(j = 2,..,n_T)$ *i.e.*, at the rest of the conditions $(T - \{t_1\})$ are:
$\xi_{g_i,t_1} = \begin{cases} 1 & \text{if } \varepsilon_{g_i,t_1} > 0 \\ 0 & \text{if } \varepsilon_{g_i,t_1} = 0 \\ -1 & \text{if } \varepsilon_{g_i,t_1} < 0 \end{cases}$	$\xi_{g_i,t_{j+1}} = \begin{cases} 1 & \text{if } \varepsilon_{g_i,t_j} < \varepsilon_{g_i,t_{j+1}} \\ 0 & \text{if } \varepsilon_{g_i,t_j} = \varepsilon_{g_i,t_{j+1}} \\ -1 & \text{if } \varepsilon_{g_i,t_j} > \varepsilon_{g_i,t_{j+1}} \end{cases}$

We see in the above discretization process that the first condition, t_1, is treated as a special case and it's discretized value is directly based on ε_{g_i,t_1} i.e., the expression value at condition t_1. For the rest of the conditions the discretized regulation value is calculated by comparing its expression value with that of the previous expression value. This helps in finding whether the gene is up- (1) or down- (-1) regulated at that particular condition. Each gene will now have a regulation pattern ($\wp$) of 0, 1, and -1 across the conditions or time points and is represented as a string. Next, *GeneClusTree* computes the angle information condition-wise for each of the gene profile based on their normalized expression values. The angle information gives the trend angle between each pair of conditions. We illustrate the whole gene-condition space as a graph with conditions across x-axis and expression levels along y-axis. Let the x-axis for a gene g_i be denoted as x_{t_j} for condition t_j and its corresponding y-axis be denoted by the expression value ε_{g_i,t_j}. Now, for the gene g_i, the angle information for each of its n_T conditions is computed according to the formula[1] given in Equation below.

$$\alpha_{\varepsilon_{g_i,t_j},\varepsilon_{g_i,t_{j+1}}} = arccos\left(\frac{\varepsilon_{g_i,t_{j+1}} - \varepsilon_{g_i,t_j}}{\sqrt{(\varepsilon_{g_i,t_{j+1}} - \varepsilon_{g_i,t_j})^2 + (x_{t_{j+1}} - x_{t_j})^2}}\right)$$

[1] Available in http://mathematics.learnhub.com/lesson/5945-trigonometry-basics

Each gene g_i will now have a pattern of angle information α_{g_i} consisting of n_T values. This angle information is then further discretized by dividing it into discrete equal intervals depending on their angle values where the width of the interval is a user input. After discretization of the angle values, each gene, g_i, will have a pattern of *angle_id*s (α'_{g_i}) across conditions, the *angle_id* value at the k^{th} condition is denoted as α'_{g_i,t_k}. The regulation information and discretized angle patterns are used together to cluster the gene expression dataset using a tree-based density approach. A string matching approach is used for matching the regulation and the discretized angle patterns of two genes. Next, we give some definitions which provide the foundation of *GeneClusTree*.

Definition 1. *Matched Subspace: The matched subspace ($M(g_i, g_j)$) is the set of k conditions where both the genes g_i and g_j match and $\theta \leq k \leq n_T$ and θ is a user defined parameter.*

Definition 2. *Maximal Matched Subspace: A matched subspace is called maximal if no superset of this can be found to be a matched subspace.*

Definition 3. *Neighbor of a gene: A gene g_j is said to be a neighbor of gene g_i i.e., $g_j \in N_l(g_i)$, iff*

i. *g_i and g_j are in the same level, say, l,*
ii. *$\mid M(g_i, g_j) \mid \geq \theta$, and*
iii. *$\alpha'_{g_j,t_k} = [\alpha'_{g_i,t_k} + \delta, \alpha'_{g_i,t_k} - \delta]$, where t_k refers to k conditions and δ has an initial value of 1 in the first level for each subtree and is incremented by 1 in every subsequent levels.*

For regulation matching, *GeneClusTree* initially attempts to find neighbors of a gene g_i over full set of conditions i.e., n_T number of conditions. If no match is found, the number of conditions is decremented by 1 at each step upto a certain threshold (say, θ) till a match occurs i.e., $\wp(g_i) \approx \wp(g_j)$. However, at each subsequent step, the previously computed matching information is used which makes the searching more efficient.

Definition 4. *Initiator: A gene g_i in the l^{th} level is said to be an initiator if $\mid N_l(g_i) \mid \geq \sigma$.*

The neighborhood of a gene g_i is searched for genes satisfying the *initiator* condition. If no neighbor gene is found, then the process is repeated with another unclassified gene. In our experiments we have obtained good results for $\sigma = 2$.

Definition 5. *Node: A node n_i in the l^{th} level is a non-empty subset of genes of G where, any gene $g_j \in n_i$ is either*
(i) itself an initiator gene, or
(ii) is within the neighborhood of an initiator gene $g_i \in n_i$ i.e., $g_j \in N_l(g_i)$.

Definition 6. *Node Reference Vector: Reference Vector of a node is the subset of conditions where all the genes belonging to that node match maximally.*

The Reference Vector of a node n_i (RV_{n_i}) to which, say, genes g_i and g_j belong to is computed as follows:

$$RV_{n_i} = \begin{cases} 1 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 1 \\ 0 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 0 \\ -1 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = -1 \\ x & \text{otherwise} \end{cases}$$

Definition 7. *Intra-node reachability: A pair of genes (g_i, g_j) in any level, say l, is said to be intra-node reachable if,*
(i) one of them is an initiator and the other is a neighbor of it, or
(ii) another gene g_k is an initiator and $g_i, g_j \in N_l(g_k)$, or
(iii) both g_i, g_j are initiators and they are neighbors to each other i.e., either $g_i \in N_l(g_j)$ or $g_j \in N_l(g_i)$.

The intra-node reachable genes satisfy the condition that they match in the same subset of conditions of regulation pattern.

Definition 8. *Inter-node Reachability: A gene $g_j \in n_c$ is said to be inter-node reachable from another gene $g_i \in n_p$, where n_p is the parent of n_c, if $\wp_{g_j}$ matches with $\wp_{g_i}$ in a total of $(n_T - (l_c - 1))$ number of conditions, where, l_c is the level of n_c.*

Finding subspace clusters in different levels gives different level of finer clustering of a dataset which may be useful for the biologists.

Definition 9. *Maximal-Space cluster: A node n_i is said to be maximal-space cluster if n_i is created at the first level (i.e., level 1) and the set of genes in n_i match in a set of k conditions, where $\theta \leq k \leq n_T$.*

Definition 10. *Reduced-Space cluster: A node n_i is a said to be a reduced-space cluster if n_i is created in the j^{th} level and the set of genes in n_i match in a set of $(k - (j - 1))$ conditions where k is cardinality of the set of conditions in which the genes in the parent node of n_i match in the $(j - 1)^{th}$ level where $2 \leq j \leq (k - (\theta - 1))$.*

Definition 11. *Noise Genes: Let $n_1, n_2, \cdots n_p$ be the set of subspace clusters of G, then noise genes in G is the set of genes not belonging to any subspace cluster n_i, i.e., noise $= \{g_x \in G \mid \forall i : g_x \notin n_i\}$*

In the rest of the paper, we will use the terms *node* and *subspace cluster* interchangeably to represent a cluster over a subset of conditions.

GeneClusTree starts by creating a tree structure in a depth-first manner with an empty node as the root. The root is at level 0 and is connected to all the nodes in level 1. The nodes in level 1 are created by a density based approach and each of these nodes is the basis of formation of the reduced-space clusters of the subtree. The process of creating a level 1 node i.e., a maximal space cluster starts

with an arbitrary unclassified gene g_i and the neighborhood of g_i is searched to check whether it is an initiator. If no gene is found to satisfy the neighborhood condition with g_i, then the process restarts with another unclassified gene. On the other hand, if g_i is an initiator gene it initiates the process of creating a new node with a node reference vector formed according to Definition 6. Then the process proceeds with finding all the genes that satisfies the intra-node reachability condition with g_i in terms of the node reference vector and are assigned to the same node to which g_i belongs. If any gene g_j from the set of intra-node reachable genes of g_i satisfies the *initiator* gene condition, then the node expansion proceeds with the gene g_j. The process continues till no more genes can be assigned to the node. Each of the nodes in level 1 is a maximal-space cluster and determines the nodes to be formed as reduced-space clusters, across different subset of conditions, in the next level of the sub-tree. After completion of the formation of the node(s) of a particular level in a sub-tree, the value of level is incremented by 1. Each of the nodes formed at level 1 becomes the parent node of the sub-trees formed at the next level (*i.e.*, level 2). Similarly, for the nodes in level i, their parents will be in level (i - 1). Also, with the increase in the height of the tree, the cardinality of the matched condition set decreases from parent to child by 1 at each level. For a particular sub-tree, the genes in each of the i^{th} nodes agrees over a set of $(k - (l - 1))$ conditions of the parent node's reference vector. Genes belonging to the sibling node(s) at the same level in a particular sub-tree have the same cardinality of matched conditions, however the match is over different set(s) of conditions. On completion of the child nodes along with their sibling nodes of a particular node in a sub-tree, the process continues similarly in the next level until the sub-tree reaches a depth of θ or no more nodes can be added to the sub-tree. The process then backtracks to level 1 and finds the next maximal subspace cluster and inserts it as a child of the root. The sub-tree of this node is created in the similar manner as described before. The whole process repeats itself until no more maximal subspace clusters are inserted in level 1 of the tree. All the remaining unclassified genes are treated as noise genes. An implementation of *GeneClusTree* is available at: http://202.141.129.18/d̃kb/resources/gene_clus_tree.rar. The following lemmas are formulated from the definitions of *GeneClusTree*.

Lemma 1. *A gene g_i belonging to $n_{1,i}$ ($n_{1,i}$ is the i^{th} subspace cluster of level 1) cannot be a neighbor of any gene $g_j \in n_{1,j}$, where $n_{1,j}$ is the j^{th} subspace cluster of level 1.*

Proof. Suppose, $g_i \in n_{1,i}$ and $g_j \in n_{1,j}$ and let $g_i \in N_l(g_j)$. Then, g_i is intra-node reachable from g_j according to Definition 7. Therefore, both g_i and g_j belongs to the same node (subspace cluster). Therefore, we come to a contradiction and hence the proof.

Lemma 2. *A gene g_j belonging to $n_{i,j}$ may not belong to $n_{(i-1),k}$ ($i = 2, 3, \cdots, \theta$) where $n_{i,j}$ refers to j^{th} subspace cluster of level i.*

Proof. Let $g_j \in n_{i,j}$, then according to Definition 8, g_j is inter node reachable from any node $n_{i-1,k}$ in level $(i-1)$ and $g_k \in n_{i-1,k}$. Then $\wp(g_j)$ matches in

a total of $(n_T - ((i-1)-1))$ *conditions, i.e., one condition less than* g_k*. The reference vector of* $n_{i,j}$ *will be same as that for* $n_{i-1,k}$ *except for the condition where* g_k *and* g_j *do not match. Therefore,* $g_j \notin n_{i-1,k}$ *and hence the proof.*

Lemma 3. *If* $g_i \in n_i$ *and* $g_j \in n_j$ *where* n_i *and* n_j *are two subspace clusters, then* g_i *and* g_j *are not intra-node reachable.*

Proof. Let g_i *and* g_j *be two intra-node reachable genes and* $g_i \in n_i$ *and* $g_j \in n_j$*, where* n_i *and* n_j *are two subspace clusters at any level. Then, according to Definition 7, either they are neighbors or both of them are neighbors of another initiator gene, that is,* g_i *and* g_j *must be in the same subspace cluster according to Definition 5. Therefore,* g_i *and* g_j *are not intra-node reachable.*

Lemma 4. *Assume* $g_i \in n_i$ *and* $g_j \in n_j$ *where* n_i *and* n_j *are two subspace clusters (*n_i *and* n_j *are not parent-child or vice versa), then* g_i *and* g_j *are not inter-node reachable.*

Proof. Let g_i *and* g_j *be two inter-node reachable genes and* $g_i \in n_i$ *and* $g_j \in n_j$*, where* n_i *and* n_j *are two subspace clusters but do not share a parent-child relationship between them. However, according to Definition 8, two genes are inter-node reachable if one of them belongs to a parent node and the other to its child. Therefore,* n_i *and* n_j *should be parent-child or vice versa and hence, are not inter-node reachable.*

Theorem 1. *Two genes* g_i *and* g_j *belonging to two different nodes are not coherent.*

Proof. Let $g_i \in n_i$ *and* $g_j \in n_j$*. Now, as per lemma 1,* g_i *cannot be a neighbor of* g_j*. Again, since genes* g_i *and* g_j *are not neighbors, then the conditions given in Definition 2 do not satisfy and hence they are not coherent.*

Theorem 2. *A gene* g_i *without a neighbor is a noise.*

Proof. A gene g_i *without a neighbor is neither intra-node nor inter-node reachable from any other node, hence such a gene* g_i *can be trivially proved to be a noise gene according to Definition 11, lemma 3 and lemma 4.*

Complexity of *GeneClusTree* in average case is $O(B \times (n_T \times n_{c_{avg}}) \times log(level_{max}))$, where B is the total number of branches in the tree and $n_{c_{avg}}$ is the average number of genes corresponding to a branch and $level_{max}$ is the maximum depth of the tree.

3 Performance Evaluation

GeneClusTree was implemented in Java in Windows environment and evaluated with several real-life datasets. Of the various datasets, the results of some of the datasets as reported in Table 1 are presented next. All the datasets are normalized to have mean 0 and standard deviation 1. The clusters formed from

Table 1. Datasets used for evaluating *GeneClusTree*

Serial No.	Dataset	No. of genes	No of conditions	Source
1	Yeast CDC28-13 [26]	384	17	http://faculty.washington.edu/kayee/cluster
2	Yeast Diauxic Shift [27]	6089	7	http://www.ncbi.nlm.nih.gov/geo/ query
3	Subset of Human Fibroblasts Serum [28]	517	13	http://www.sciencemag.org/feature/data/984559.hsl

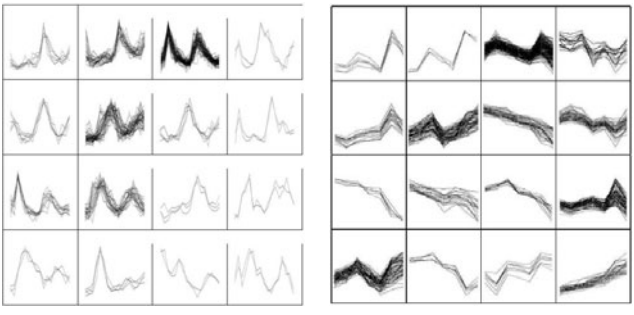


Fig. 1. a) Some of the clusters from the Dataset 1 b) Some of the clusters obtained from Dataset 2

Dataset 1 and 2 are shown in Figure 1 (a) and (b). Figure 3 (a), shows some of the maximal space and reduced space clusters of Dataset 1. The maximal and reduced space clusters of Dataset 3 reported in [28] are shown in Figure 3 (b). In order to validate our clustering result, we employ z-score [29] as the measure of agreement.

Z-score [29] is calculated by investigating the relation between a clustering result and the functional annotation of the genes in the cluster. We have used Gibbons ClusterJudge [29] tool to calculate the z-score. A higher value of z indicates that genes would be better clustered by function, indicating a more biologically relevant clustering result. To test the performance of the clustering algorithm, we compare clusters identified by *GeneClusTree* with the results from RDClust, DCCA and UPGMA. In this paper, the reported z-score is averaged over 10 repeated experiments. The result of applying the z-score on Dataset 1 is shown in Table 2. In this table the proposed algorithm has been compared with the well known agglomerative hierarchical algorithm, UPGMA, DCCA and RDCLust. Table 2 clearly shows that *GeneClusTree* outperforms UPGMA, RDClust and DCCA w.r.t. the cluster quality. We note here that unlike k-means our method does not require the number of clusters as an input parameter. It detects the clusters present in the dataset automatically and gives the rest as noise. However, the algorithm UPGMA requires the input parameter $cutoff$.

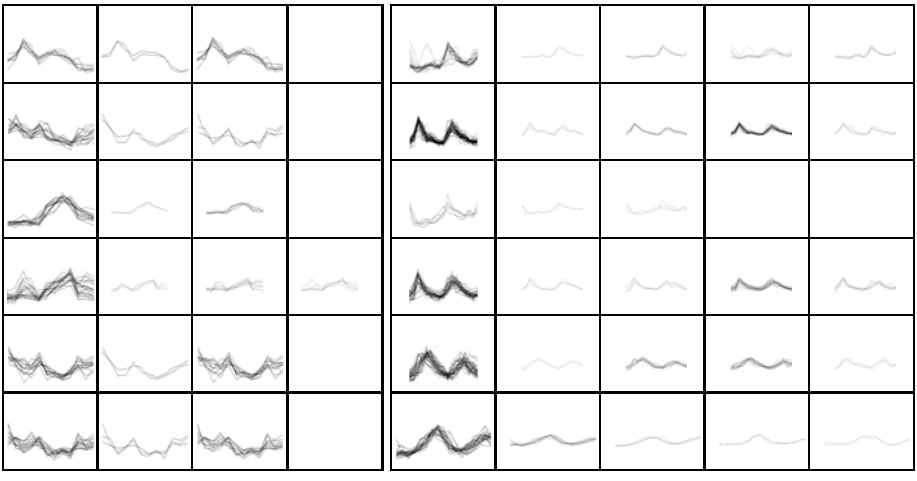


Fig. 2. (a) Each of the rows represents the six clusters formed from Dataset 1. Starting from the second column of each row, the reduced space clusters are illustrated for the maximal space cluster given in the first column. (b) Each of the rows represents some of the clusters formed from Dataset 3. Starting from the second column of each row, the reduced space clusters are illustrated for the maximal space cluster given in the first column.

The **biological relevance** of a cluster can be verified based on the gene ontology (GO) annotation database *http://db.yeastgenome.org/cgi-bin/GO/goTermFinder*. It is used to test the functional enrichment of a group of genes in terms of three structured controlled ontologies, *viz.*, associated biological processes, molecular functions and biological components. The functional enrichment of each GO category in each of the clusters obtained is calculated by its *p-value* [30]. A low p-value indicates the genes belonging to the enriched functional categories are biologically significant in the corresponding clusters. To compute the p-value, we used the software FuncAssociate [30]. Due to the constraint in the page limit, the enriched functional categories for some of the clusters obtained by *GeneClusTree* on Dataset 1 are partially listed in Table 3. The functional enrichment of each GO category in each of the clusters is calculated by its p-value. Cluster C2 contains genes involved in DNA replication with the highly enriched category being 'MCM complex' with a p-value of 1.1×10^{-12}. The highly enriched categories in C5 is the 'cellular bud' with a p-value of 7.0×10^{-07}. The genes in cluster C10 are involved in cell cycle. C10 contains the highly enriched cellular components of 'DNA metabolic process', 'DNA replication', 'chromosome', 'chromosomal part', 'cell cycle, etc. with p-values of 1.8×10^{-22}, 1.8×10^{-21}, 9.7×10^{-21}, 1.5×10^{-20}

Table 2. z-scores for *GeneClusTree* and its counterparts for Dataset 1

	UPGMA	SOM	CAST	ECAST	DCCA	RDClust	GCE	*GeneClusTree*
No. of clusters	16	09	11	42	10	10	15	17
z-score	5.57	3.2	2.48	3.97	6.2	6.95	6.37	7.42

Table 3. *P*-value of some of the clusters of Dataset 1

Cluster	P-value	GO number	GO category
C2	1.1e-12	GO:0042555	MCM complex
	5.2e-10	GO:0005656	pre-replicative complex
	5.2e-10	GO:0006267	pre-replicative complex assembly
	1.5e-09	GO:0000084	S phase of mitotic cell cycle
	3.5e-09	GO:0031261	DNA replication preinitiation complex
	3.5e-09	GO:0043596	nuclear replication fork
	3.5e-09	GO:0005739	S phase
	1.1e-08	GO:0044455	DNA replication origin binding
	3.8e-08	GO:0051329	interphase of mitotic cell cycle
	4.1e-08	GO:0051325	interphase
	5.4e-08	GO:0005657	replication fork
	6.3e-08	GO:0006270	DNA replication initiation
	7.2e-08	GO:0008094	DNA-dependent ATPase activity
	1.1e-07	GO:0009378	four-way junction helicase activity
	1.2e-07	GO:0022403	cell cycle phase
	3.9e-07	GO:0022402	cell cycle process
C5	7e-07	GO:0005933	cellular bud
	5.1e-06	GO:0004857	enzyme inhibitor activity
	8.3e-06	GO:0004860	protein kinase inhibitor activity
	8.3e-06	GO:0019210	kinase inhibitor activity
	9.6e-06	GO:0030427	site of polarized growth
	2.8e-05	GO:0005935	cellular bud neck
	4.7e-05	GO:0019887	protein kinase regulator activity
	5.6e-05	GO:0019207	kinase regulator activity
	6.7e-05	GO:0004861	cyclin-dependent protein kinase inhibitor activity
C10	1.8e-22	GO:0006259	DNA metabolic process
	1.8e-21	GO:0006260	DNA replication
	9.7e-21	GO:0005694	chromosome
	1.5e-20	GO:0044427	chromosomal part
	8.8e-19	GO:0007049	cell cycle
	1.4e-18	GO:0006281	DNA repair
	1.3e-16	GO:0006974	response to DNA damage stimulus
	4.7e-16	GO:0009719	response to endogenous stimulus
	6.4e-16	GO:0006261	DNA-dependent DNA replication
	1e-05	GO:0006261	DNA-dependent DNA replication
	5.6e-14	GO:0022402	cell cycle process
	9.6e-14	GO:0005634	nucleus
	1e-13	GO:0007064	mitotic sister chromatid cohesion
	7.2e-13	GO:0005657	replication fork
	9.1e-13	GO:0022403	cell cycle phase
	3e-12	GO:0051276	chromosome organization and biogenesis
	3.5e-12	GO:0000228	nuclear chromosome
	4.4e-12	GO:0000278	mitotic cell cycle
	6.4e-12	GO:0007062	sister chromatid cohesion
	1.2e-11	GO:0044454	nuclear chromosome part
	7.5e-11	GO:0006273	lagging strand elongation
	3.1e-10	GO:0043228	non-membrane-bounded organelle
	3.1e-10	GO:0043232	intracellular non-membrane-bounded organelle
	4.2e-10	GO:0006271	DNA strand elongation during DNA replication
	4.2e-10	GO:0022616	DNA strand elongation
	5.1e-10	GO:0030894	replisome
	5.1e-10	GO:0043601	nuclear replisome
	5.7e-10	GO:0006950	response to stress
	9.1e-10	GO:0051052	regulation of DNA metabolic process
	1.1e-09	GO:0000819	sister chromatid segregation
	1.5e-09	GO:0006139	nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	6.5e-09	GO:0045934	negative regulation of nucleobase, nucleoside, nucleotide and nucleic acid metabolic process
	6.5e-09	GO:0000070	mitotic sister chromatid segregation
	8.6e-09	GO:0043596	nuclear replication fork
	3.1e-08	GO:0051301	cell division
	3.8e-08	GO:0000793	condensed chromosome
	5.4e-08	GO:0031324	negative regulation of cellular metabolic process
	5.4e-08	GO:0003677	DNA binding
	5.7e-08	GO:0009892	negative regulation of metabolic process
	7.8e-08	GO:0000279	M phase

and 8.8×10^{-19} being the highly enriched one. To restrict the size of the article we have reported the p-values of only three clusters obtained from Dataset 1. From the Table 3, we can conclude that *GeneClusTree* shows a good enrichment of functional categories and therefore project a good biological significance.

4 Conclusions and Future Work

This work presents a tree-based density approach which finds useful subgroups of genes within a cluster and obtains a tree structure of the dataset where the clusters at the top level gives the finer clustering of the dataset. *GeneClusTree* does not require the number of clusters apriori and the clusters obtained have been found satisfactory on visual inspection and also based on z-score as well as p-values for three real datasets. However, work is going on for establishing the effectiveness of *GeneClusTree* over more real-life human gene datasets. In our future work, we plan to fuse the analysis of gene expression datasets with biological information during node expansion to identify the co-regulated genes.

Acknowledgement. This work is an outcome of the research project in collaboration with CSCR, ISI, Kolkata funded by DST, Govt. of India.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Fransisco (2004)
2. Stekel, D.: Microarray Bioinformatics. Cambridge University Press, Cambridge (2006)
3. Ben-Dor, A., Shamir, R., Yakhini, Z.: Clustering gene expression patterns. Journal of Computational Biology 6(3-4), 281–297 (1999)
4. Eisen, M., Spellman, P., Brown, P., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. of National Academy of Sciences 95, 14863–14868 (1998)
5. Jiang, D., Tang, C., Zhang, A.: Cluster analysis for gene expression data: A survey (2003), www.cse.buffalo.edu/DBGROUP/bioinformatics/papers/survey.pdf
6. Jiang, D., Pei, J., Zhang, A.: DHC: a density-based hierarchical clustering method for time series gene expression data. In: Proc. of BIBE, Bethesda, Maryland, p. 393 (2003)
7. McQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
8. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
9. Heyer, L., Kruglyak, S., Yooseph, S.: Exploring expression data: identication and analysis of co-expressed genes. Genome Research 9, 1102–1115 (1999)
10. Tamayo, P., Slonim, D., Mesirov, J., et al.: Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proc. of National Academy of Sciences 96(6), 2907–2912 (1999)

11. Dopazo, J., Carazo, J.M.: Phylogenetic reconstruction using an unsupervised neural network that adopts the topology of a phylogenetic tree. Journal of Molecular Evolution 44, 226–233 (1997)
12. Bhattacharya, A., De, R.: Divisive correlation clustering algorithm (DCCA) for grouping of genes: detecting varying patterns in expression profiles. Bioinformatics 24(11), 1359–1366 (2008)
13. Luo, F., Khan, L., Bastani, F., et al.: A dynamically growing self-organizing tree (DGSOT) for hierarchical clustering gene expression profiles. Bioinformatics 20(16), 2605–2617 (2004)
14. Shu, G., Zeng, B., Chen, Y.P., Smith, O.H.: Performance assessment of kernel density clustering for gene expression profile data. Comparative and Functional Genomics 4, 287–299 (2003)
15. Das, R., Bhattacharyya, D.K., Kalita, J.K.: Clustering gene expression data using a regulation based density clustering. IJRTE 2(1-6), 76–78 (2009)
16. Das, R., Bhattacharyya, D.K., Kalita, J.K.: Clustering gene expression data using an effective dissimilarity measure. IJCB (Special Issue) 1(1), 55–68 (2010)
17. Jarvis, R.A., Patrick, E.A.: Clustering using a similarity measure based on shared nearest neighbors. IEEE Transactions on Computers 11 (1973)
18. Herrero, J., Valencia, A., Dopazo, J.: A hierarchical unsupervised growing neural network for clustering gene expression patterns. Bioinformatics 17, 126–136 (2001)
19. Sharan, R., Shamir, R.: CLICK: A clustering algorithm with applications to gene expression analysis. In: Proc. of ISMB. AAAI Press, Menlo Park (2000)
20. Bellaachia, A., Portnoy, D., Chen, A.G., Elkahloun, Y.: E-CAST: A data mining algorithm for gene expression data. In: Proc. of the BIOKDD, vol. 49 (2002)
21. Das, R., Kalita, J.K., Bhattacharyya, D.K.: A new approach for clustering gene expression time series data. IJBRA 5(3), 310–328 (2009)
22. Priyadarshini, G., Chakraborty, B., Das, R., et al.: Highly coherent pattern identification using graph-based clustering. In: Proc. of BIOT, Lafayette, Louisiana, US, pp. 29–38 (2010)
23. Bandyopadhyay, S., Mukhopadhyay, A., Maulik, U.: An improved algorithm for clustering gene expression data. Bioinformatics 23(21), 2859–2865 (2007)
24. Maulik, U., Mukhopadhyay, A., Bandyopadhyay, S.: Combining pareto-optimal clusters using supervised learning for identifying co-expressed genes. BMC Bioinformatics 10(27) (2009)
25. Yun, T., Hwang, T., Cha, K., et al.: CLIC: clustering analysis of large microarray datasets with individual dimension-based clustering. Nucleic Acids Research 38, W246–W253 (2010)
26. Cho, R.J., Campbell, M., Winzeler, E., et al.: A genome-wide transcriptional analysis of the mitotic cell cycle. Molecular Cell 2(1), 65–73 (1998)
27. DeRisi, J.L., Iyer, V.R., Brown, P.O.: Exploring the metabolic and genetic control of gene expression on a genomic scale. Science 278, 680–686 (1997)
28. Iyer, V.R., Eisen, M.B., Ross, D.T., et al.: The transcriptional program in the response of the human fibroblasts to serum. Science 283, 83–87 (1999)
29. Gibbons, F., Roth, F.: Judging the quality of gene expression based clustering methods using gene annotation. Genome Research 12, 1574–1581 (2002)
30. Berriz, F.G., et al.: Characterizing gene sets with funcassociate. Bioinformatics 19, 2502–2504 (2003)

Nonlinear Discriminative Embedding for Clustering via Spectral Regularization *

Yubin Zhan and Jianping Yin

School of Computer, National University of Defense Technology, Changsha, 410073, P.R. China
{Yubinzhan,JPYin}@nudt.edu.cn

Abstract. In this paper, we propose a novel nonlinear discriminative dimensionality reduction method for clustering high dimensional data. The proposed method first represents the desired low dimensional nonlinear embedding as linear combinations of predefined smooth vectors with respect to data manifold, which are characterized by a weighted graph. Then the optimal combination coefficients and optimal cluster assignment matrix are computed by maximizing between-cluster scatter and minimizing within-cluster scatter simultaneously as well as preserving smoothness of the cluster assignment matrix with respect to the data manifold. We solve the optimization problem in an iterative algorithm, which is proved to be convergent. The contributions of the proposed method are two folds: 1) obtained nonlinear embedding can recover intrinsic manifold structure as well as enhance the cluster structure of the original data; 2) the cluster results can also be obtained in dimensionality reduction procedure. Extensive experiments conducted on UCI data sets and real world data sets have shown the effectiveness of the proposed method for both clustering and visualization high dimensional data set.

Keywords: dimensionality reduction; cluster; Laplacian graph; spectral regularization.

1 Introduction

Real applications in many domains such as pattern recognition, computer vision and data mining often lead to very high dimensional data. Dimensionality Reduction (DR) is an effective and widely used approach to deal with such high dimensional data, due to its potential of mitigating the so-called "curse of dimensionality". Up to now, many dimensionality reduction methods (DRs) have been developed, such as Principal Component Analysis (PCA) [1], Isomap [2], Locally Linear Embedding (LLE) [3], Laplacian Eigenmap (LE) [4] and supervised Linear Discriminant Analysis (LDA) [5].

Although existing unsupervised DRs have different motivations and concrete implementations, they obtain the desired low dimensional coordinates by preserving a certain property of original data, whether global or local. For example, property that PCA preserves is the global variance and Isomap aims to preserve the geometric distance on the

* This work is supported by National Natural Science Foundation of China (Grant No. 60970034, 60603015), and the Foundation for Author of National Excellent Doctoral Dissertation(Grant No. 2007B4).

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 237–248, 2011.

intrinsic manifold. While some other approaches try to preserve a certain local property of data, such as locality that LE tries to preserve and neighborhood relationship that LLE aims to preserve.

However, cluster structure as a key property of data, which reflects intrinsic distribution of data and plays a crucial role in further analyzing and utilizing data [6], has been ignored by these popular DRs. Here the term "cluster structure" means the natural aggregation of similar objects and natural separation of dissimilar objects in concept level. Hence, it is appealing to devise DR method for clustering that can preserve or enhance cluster structure of the original high dimensional data in the transformed low dimensional embedded space.

On the other hand, due to sparsity of data in high dimensional space, clustering directly on high dimensional data is still a challenging problem. A natural solution is to transform data into a low dimensional compact space through aforementioned DRs such as PCA before clustering. However, due to the intrinsic gap between clustering and existing DRs, which are not designed originally for clustering, clustering structure of original data can't be well preserved and may be even destroyed in the transformed low dimensional space.

Therefore, in this paper, a novel DR approach for clustering is proposed. In the proposed method, both nonlinear DR and clustering task can be achieved simultaneously. Firstly, according to graph embedding theory [7], a weighted graph which can characterize manifold structure of the whole data is constructed. Then the desired low dimensional coordinates are represented as linear combinations of smooth functions defined on data graph, which are the eigenvectors corresponding to the smallest eigenvalues of the Laplacian matrix of data graph. The key idea behind this is that the learned low dimensional coordinates should be as smooth as possible with respect to the data manifold. Finally optimal combination coefficients and cluster assignment matrix can be computed by maximizing between-cluster scatter and minimizing within-cluster scatter simultaneously as well as preserving smoothness of cluster assignment matrix with respect to data manifold.

2 Spectral Regularization on Manifold

Given a data set of n objects $\mathcal{X} = \{x_i\}_{i=1}^n$, first a weighted graph $G(\mathcal{X}, W)$ can be constructed, where the (i, j) entry w_{ij} of affinity matrix W measures similarity between samples x_i and x_j. There are various similarity criteria. In this paper, the popular Gaussian similarity as Ref. [8] is adopted.

According to [9], smoothness of a function $f : \mathcal{X} \to R$ on graph G can be measured by:

$$\Omega(f) = \frac{1}{2} \sum_{i,j} w_{ij} (f(x_i) - f(x_j))^2 \tag{1}$$

The smaller $\Omega(f)$ is, the smoother f is. This measure penalizes large changes over data points that have large similarities. It is the same as manifold assumption that nearby points are likely to have the same labels.

Let $L = D - W$ be the Laplacian matrix of graph G, where D is diagonal matrix with element $d_{ii} = \sum_j w_{ij}$. Then the smoothness of function f defined in Eq. (1) can be also expressed as:

$$\Omega(f) = \mathbf{f}^{\mathrm{T}} L \mathbf{f} \tag{2}$$

where $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))^{\mathrm{T}}$. One can also use the normalized smoothness measure $\bar{L} = D^{-1/2} L D^{-1/2}$ instead of L in Eq. (2). Note that any function f on graph G is identical to an n-dimensional vector $\mathbf{f}$. Thus one can also regard an n-dimensional vector as a function on graph G.

It is noteworthy to study the smoothness of the eigenvectors of matrix L. Let (λ_i, v_i) be the pair of eigenvalue and eigenvector of L, where $0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and v_i has been normalized to 1. Then smoothness of v_i is:

$$\Omega(v_i) = v_i^{\mathrm{T}} L v_i = \lambda_i \tag{3}$$

This means that eigenvalue λ_i measures smoothness of eigenvector v_i: the smaller λ_i is, the smoother v_i is.

From definition of L, one can see that L is positive semidefinite. Thus these n eigenvectors of L are mutual orthogonal and form the orthogonal basis of function space on graph G. Then for any function $\mathbf{f}$ on G, it can be expressed as:

$$\mathbf{f} = \sum_{i=1}^{n} a_i v_i \tag{4}$$

Note that $\lambda_1 = 0$ and $v_1 = e/\sqrt{n}$, where e is all 1's vector with suitable dimensionality. So v_1 is a trivial function on graph G. Then we rewrite Eq. (4) as:

$$\mathbf{f} = a_1 v_1 + \sum_{i=1}^{m} a_{i+1} v_{i+1} + \sum_{i=m+2}^{n} a_i v_i = T_0 + T_1 + T_2 \tag{5}$$

where $m \ll n$. The T_0, T_1 and T_2 are called the trivialness component, smoothness component and noise component of function $\mathbf{f}$, respectively. Discarding noise component T_2 of function $\mathbf{f}$ is to improve the smoothness of function and discarding trivialness component T_0 is just to translate function and has no essential effect. Hence representing a function as T_1 term in fact imposes smoothness assumption on function. This regularization technique is called spectral regularization [10]. Here we also introduce the trivialness component T_0, which is a little different from the original spectral regularization proposed in Ref. [10]. The convenience of this variation will be given in the following section.

3 The Proposed Method

3.1 Problem Formulation and Its Solution

Our main goal is to cluster the given data set $\mathcal{X} = \{x_i\}_{i=1}^n$ into K clusters $C_i (i = 1, 2, \ldots, K)$ in DR procedure. Denote the cluster assignment matrix by $P = (p_{ij}) \in$

$\mathbb{B}^{n\times K}$ where $p_{ij} = 1$ if $x_i \in C_j$ and otherwise $p_{ij} = 0$. A scaled cluster assignment matrix $\hat{P} = (\hat{p}_{ij})$, where $\hat{p}_{ij} = p_{ij}/\sqrt{n_j}$ and n_j is the number of samples in cluster C_j, can also be introduced. It is easy to check that $\hat{P} = P(P^{\mathrm{T}}P)^{-1/2}$ and $\hat{P}^{\mathrm{T}}\hat{P} = I_K$ where I_K is the $K \times K$ identity matrix.

Let $F = (\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_d)^{\mathrm{T}}$ be the desired low dimensional coordinates of $\mathcal{X}$ where the ith column vector is the new low dimensional representation of sample x_i. According to manifold assumption, i.e., nearby points should be also close to each other in the low dimensional space, for each vector $\mathbf{f}_i$ in F, we would like it to be a smooth function on the data graph G. This implies that we can only use the smoothness component T_1 to represent $\mathbf{f}_i$, i.e., $\mathbf{f}_i$ can be expressed as a linear combination of predefined smooth function of graph G:

$$\mathbf{f}_i = \sum_{j=1}^{m} a_{ji} v_{j+1} = V_m a_i \tag{6}$$

where $V_m = (v_2, v_3, \ldots, v_{m+1})$. Then $F = (V_m A)^{\mathrm{T}}$ where $A = (a_1, a_2, \ldots, a_d)$.

Since $v_i^{\mathrm{T}} e = 0 (i = 2, 3, \ldots, n)$, we have $Fe = (V_m A)^{\mathrm{T}} e = 0$, which implies that in new low dimensional space, the samples are automatically centered at origin. This property can significantly simplify computation of total scatter matrix S_t and between-cluster scatter matrix S_b in the transformed space, which can be expressed as:

$$S_t = FF^{\mathrm{T}} = (V_m A)^{\mathrm{T}}(V_m A) = A^{\mathrm{T}} A; \quad S_b = F\hat{P}\hat{P}^{\mathrm{T}}F^{\mathrm{T}} = A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}\hat{P}^{\mathrm{T}} V_m A$$

As stated in section 1, besides obtaining low dimensional embedding, the main goal of our method is to preserve or even enhance cluster structure of data in low dimensional space. This is implemented by maximizing between-cluster scatter and minimizing within-cluster scatter simultaneously:

$$\max : \frac{\mathrm{tr}\,(S_b)}{\mathrm{tr}\,(S_t)} = \frac{\mathrm{tr}\,(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}\hat{P}^{\mathrm{T}} V_m A)}{\mathrm{tr}\,(A^{\mathrm{T}} A)} \tag{7}$$

To make the problem well defined, we impose orthogonal constraint $A^{\mathrm{T}}A = I_d$ on combination coefficient A. Hence the above trace ratio problem can be reduced to the following constrained trace maximization problem:

$$\max_{A^{\mathrm{T}}A = I_d} : \quad \mathrm{tr}\,(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}\hat{P}^{\mathrm{T}} V_m A) \tag{8}$$

Another consideration for cluster assignment matrix is that points with high similarities are much likely to have the same cluster label. Mathematically, it is equal to minimize the following regularization term on $\hat{P}$:

$$\min_{\hat{P}^{\mathrm{T}}\hat{P} = I_K} : \mathrm{tr}\,(\hat{P}^{\mathrm{T}} L \hat{P}) \tag{9}$$

Combining the objective in Eqs. (8) and (9), then we can derive the final optimization objective for the proposed method:

$$\max_{\substack{A^{\mathrm{T}}A = I_d \\ \hat{P}^{\mathrm{T}}\hat{P} = I_K}} : \mathrm{tr}\,(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}\hat{P}^{\mathrm{T}} V_m A) - \gamma\,\mathrm{tr}\,(\hat{P}^{\mathrm{T}} L \hat{P}) \tag{10}$$

where $\gamma > 0$ is a trade-off parameter to balance the two terms. Since the discrete constraint of $\hat{p}_{ij}$ makes the problem an NP-hard problem, we will relax it to be continuous value as many spectral clustering algorithms do [11].

As one can see, the objective in Eq. (10) depends on both A and $\hat{P}$, which depend on each other. We cannot give a closed-form solution. Here we optimize them in an iterative manner. In other words, we will optimize the objective with respect to one variable while fixing the other one. This procedure repeats until convergence.

For a fixed $\hat{P}$, optimal combination coefficients A can be obtained by solving the following trace maximization problem:

$$\max_{A^{\mathrm{T}}A=I_d} : \quad \mathrm{tr}\,(A^{\mathrm{T}}V_m^{\mathrm{T}}\hat{P}\hat{P}^{\mathrm{T}}V_mA) \tag{11}$$

Based on Ky Fan theorem [11], the optimal A consists of d eigenvectors of $V_m^{\mathrm{T}}\hat{P}\hat{P}^{\mathrm{T}}V_m$ corresponding to the first d largest eigenvalues, i.e., $A = (a_1, a_2, \ldots, a_d)$ where $a_i (i = 1, 2 \ldots, d)$ is the ith largest eigenvector of matrix $V_m^{\mathrm{T}}\hat{P}\hat{P}^{\mathrm{T}}V_m$.

Similarly, for a fixed A, optimal relaxed assignment matrix $\hat{P}$ can be computed by maximizing the following:

$$\max_{\hat{P}^{\mathrm{T}}\hat{P}=I_K} : \quad \mathrm{tr}\,(A^{\mathrm{T}}V_m^{\mathrm{T}}\hat{P}\hat{P}^{\mathrm{T}}V_mA) - \gamma\,\mathrm{tr}\,(\hat{P}^{\mathrm{T}}L\hat{P}) = \mathrm{tr}\,(\hat{P}^{\mathrm{T}}(V_mAA^{\mathrm{T}}V_m^{\mathrm{T}} - \gamma L)\hat{P})$$

After obtaining continuous optimal scaled cluster assignment matrix, we can use K-means or spectral rotation to obtain discrete cluster assignment matrix [12].

We refer to the proposed method as *Nonlinear Discriminative Embedding for Clustering via Spectral Regularization* (NDECSR), whose algorithm procedure are summarized in the following:

NDECSR with its iterative solution:

Input: Data matrix $X = [x_1, x_2, \ldots, x_n)]$, the number of smooth basis vectors m, trade-off parameter λ, neighborhood size k, the desired dimensionality d, the number of clusters K.
Output: low dimensional coordinates F, cluster assignment matrix P.

1: Construct weighted graph G with neighborhood size k and compute V_m
2: Cluster data into K clusters by K-means algorithm to obtain initial $\hat{P}_0$.
3: Iterative optimization: For $q = 1, 2, \ldots, q_{\max}$, do

(1). Solve eigenvalue decomposition problem:

$$V_m^{\mathrm{T}}\hat{P}_{q-1}\hat{P}_{q-1}^{\mathrm{T}}V_mu_i = \lambda_iu_i, (i = 1, 2, \ldots, d) \tag{12}$$

where u_i is the ith largest eigenvector. Set $U = [u_1, u_2, \ldots, u_d]$.

(2). Reshape U for orthogonal transformation invariance: let $S = UU^{\mathrm{T}}V_mV_m^{\mathrm{T}}UU^{\mathrm{T}}$, solve eigenvalue decomposition:

$$Sa_i = \gamma_ia_i \tag{13}$$

and let $A_q = [a_1, a_2, \ldots, a_d]$, where a_i is the ith largest eigenvector.

(3). Solve eigenvalue decomposition problem:

$$(V_mA_qA_q^{\mathrm{T}}V_m^{\mathrm{T}} - \gamma L)u_j = \lambda_j'u_j, (j = 1, 2, \ldots, K) \tag{14}$$

where u_j is the jth largest eigenvector. Set $U = [u_1, u_2, \ldots, u_K]$.

(4). Reshape U for orthogonal transformation invariance: let $S = UU^{\mathrm{T}}XX^{\mathrm{T}}UU^{\mathrm{T}}$, solve eigenvalue decomposition:

$$Sp_i = \gamma_i p_i \tag{15}$$

and let $\hat{P}_q = [p_1, \ldots, p_K]$, where p_i is the ith largest eigenvector.

if $\|A_q - A_{q-1}\|_F^2 < \varepsilon$ and $\|\hat{P}_q - \hat{P}_{q-1}^{\mathrm{T}}\|_F^2 < \varepsilon$ (we set $\varepsilon = 0.001$ in our experiments), then return.

4: Let the optimal coefficient matrix $A = A_q$ and $\hat{P} = \hat{P}_q$. Obtain the discrete cluster assignment matrix P via K-means or spectral rotation and compute low embedding F via $F = (V_m A)^{\mathrm{T}}$.

3.2 Convergence Analysis

In this subsection, we will study the convergence of our NDECSR algorithm.

Denote the objective function value in each iteration by

$$\begin{aligned} J(\hat{P}_q, A_q) &= \mathrm{tr}\,(A_q^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_q \hat{P}_q^{\mathrm{T}} V_m A_q) - \gamma\,\mathrm{tr}\,(\hat{P}_q^{\mathrm{T}} L \hat{P}_q) \\ &= \mathrm{tr}\,(\hat{P}_q^{\mathrm{T}} (V_m A_q A_q^{\mathrm{T}} V_m^{\mathrm{T}} - \gamma L) \hat{P}_q) \end{aligned} \tag{16}$$

then we have Theorem 1.

Theorem 1. *The objective function $J(\hat{P}_q, A_q)$ satisfies the following inequality:*

$$J(\hat{P}_q, A_q) \le J(\hat{P}_q, A_{q+1}) \le J(\hat{P}_{q+1}, A_{q+1}) \tag{17}$$

Proof. According to our NDECSR algorithm, $\hat{P}_{q+1}$ and A_{q+1} satisfy:

$$\begin{aligned} A_{q+1} &= \arg \max_{A^{\mathrm{T}}A = I_d} \mathrm{tr}(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_q \hat{P}_q^{\mathrm{T}} V_m A) \\ &= \arg \max_{A^{\mathrm{T}}A = I_d} \mathrm{tr}(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_q \hat{P}_q^{\mathrm{T}} V_m A) - \gamma\,\mathrm{tr}(\hat{P}_q^{\mathrm{T}} L \hat{P}_q) = \arg \max_{A^{\mathrm{T}}A = I_d} J(\hat{P}_q, A) \end{aligned} \tag{18}$$

$$\hat{P}_{q+1} = \arg \max_{\hat{P}^{\mathrm{T}}\hat{P} = I_K} \mathrm{tr}(\hat{P}^{\mathrm{T}} (V_m A_{q+1} A_{q+1}^{\mathrm{T}} V_m^{\mathrm{T}} - \gamma L) \hat{P}) = \arg \max_{\hat{P}^{\mathrm{T}}\hat{P} = I_K} J(\hat{P}, A_{q+1}) \tag{19}$$

Then we have: $J(\hat{P}_q, A_q) \le J(\hat{P}_q, A_{q+1}) \le J(\hat{P}_{q+1}, A_{q+1})$.

Theorem 1 means that the objective function monotonously increases as the iteration number q. Another fact is that the objective function has upper bound under constraints in (10). Therefore objective function will converge in limited iterations. Then we can further obtain the following Theorem which gives the convergency of A_q and $\hat{P}_q$:

Theorem 2. *The matrix sequence $\{(\hat{P}_q, A_q)\}$ obtained by NDECSR is convergent.*

Proof. From Theorem 1, we can assume there exist Q iterations when the objective function converges. Then we have:

$$J(\hat{P}_Q, A_Q) = J(\hat{P}_Q, A_{Q+1}) = J(\hat{P}_{Q+1}, A_{Q+1}) \tag{20}$$

Substituting (16) into (20) and by using (18) and (19), we can obtain that:

$$\begin{aligned}
&\mathrm{tr}(A_Q^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_Q \hat{P}_Q^{\mathrm{T}} V_m A_Q) = \mathrm{tr}(A_{Q+1}^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_Q \hat{P}_Q^{\mathrm{T}} V_m A_{Q+1}) \\
&\quad = \max_{A^{\mathrm{T}} A = I_d} \mathrm{tr}(A^{\mathrm{T}} V_m^{\mathrm{T}} \hat{P}_Q \hat{P}_Q^{\mathrm{T}} V_m A) \\
&\Longrightarrow \exists \text{ orthogonal matrix } O \text{ such that:} \quad A_{Q+1} = A_Q O \qquad (21) \\
&\mathrm{tr}(\hat{P}_{Q+1}^{\mathrm{T}} (V_m A_{Q+1} A_{Q+1}^{\mathrm{T}} V_m^{\mathrm{T}} - \gamma L) \hat{P}_{Q+1}) = \mathrm{tr}(\hat{P}_Q^{\mathrm{T}} (V_m A_{Q+1} A_{Q+1}^{\mathrm{T}} V_m^{\mathrm{T}} - \gamma L) \hat{P}_Q) \\
&\quad = \max_{\hat{P}^{\mathrm{T}} \hat{P} = I_K} \mathrm{tr}(\hat{P}^{\mathrm{T}} (V_m A_{Q+1} A_{Q+1}^{\mathrm{T}} V_m^{\mathrm{T}} - \gamma L) \hat{P}) \\
&\Longrightarrow \exists \text{ orthogonal matrix } O' \text{ such that:} \quad \hat{P}_{Q+1} = \hat{P}_Q O' \qquad (22)
\end{aligned}$$

Hence A_{Q+1} and A_Q have the same column space. So do $\hat{P}_{Q+1}$ and $\hat{P}_Q$. Note that in step 3(2) and 3(4) of our NDECSR algorithm, we have reshaped A_q and $\hat{P}_q$ for orthogonal transformation invariance. Thus $A_{Q+1} = A_Q$ and $\hat{P}_{Q+1} = \hat{P}_Q$. This means that when the objective function convergences, the obtained coefficient matrix A_q and relaxed assignment matrix $\hat{P}_q$ also converge.

4 Experimental Results

In this section, the clustering performance of NDECSR is compared with K-means clustering (KM), Normalized Cut (NCut) [13], Discriminative K-means (DKM) [14] and Spectral Embedded Clustering (SEC) [8] via systematic experiments. Moreover, the visualization results of NDECSR are also compared with two popular manifold learning algorithms: LLE [3] and LE [4].

4.1 Data Sets

Experiments are conducted on three categories of data sets, which are popular benchmark data sets covering a wide range of applications.

UCI data: Three UCI data set Ionosphere, Iris and Wine are used in our experiments.

Image data: We perform experiments on four image data sets: ORL face data set[1], MNIST data set[2], Coil-20 data set [15] and a scene category data set (Scene) [16], which are the representatives of four different image recognition problems. For ORL face data, the images are resized to 32×32. For MNIST data set we only select five digits '1','2','3','4' and '5' and 300 images per digit. The Coil-20 data set contains 20 objects and each object has 72 images with size 32×32. In the original scene category data set, there are 8 different scenes. We only select 4 different scenes including coast, forest, high way and tall building. We use the feature called Spatial Envelope [16] to represent each scene image, the feature is a 960-dimensional vector.

[1] http://www.uk.research.att.com/facedatabase.html
[2] http://yann.lecun.com/exdb/mnist/

Text data: We also perform experiments on two text data sets: 20newsgroup[3] and TDT2[4]. For 20newsgroup data set, we random choose 2380 documents from the topic rec which contains autos, motorcycles, baseball and hockey, and delete the words that occur less than 10 times in all these 2380 documents. The final TF-IDF representation of each document is a 4477-dimensional vector. For TDT2 data set, we select all documents from categories 20013, 20070, 20044 and 20076, which are the categories that contain 4th, 5th, 6th and 7th most documents, respectively. They are total 1931 documents and we also delete the words that occur less than 5 times, the final TF-IDF representation of each document is a 7906-dimensional vector.

Table 1 summarizes the details of data sets used in our experiments. For the random selected subset data, we conduct 10 random experiments to obtain the average results.

Table 1. Descriptions of the Data sets

category	Data	#Samples	#Dim	#class
UCI	Ionosphere	351	33	2
	Iris	150	4	3
	Wine	178	13	3
image	ORL	400	1024	40
	MNIST-12345	1500	784	5
	COIL20	1440	1024	20
	Scene	1304	960	4
text	20NG_rec	2380	4477	4
	TDT2	1931	7906	4

Table 2. Comparison of Clustering Accuracy of different methods

	KM	NCut	DKM	SEC	NDECSR
Ionosphere	0.712	0.715	0.715	0.725	**0.849**
Iris	0.855	0.946	0.876	0.946	**0.950**
Wine	0.953	0.966	0.944	0.972	**0.978**
ORL	0.554	0.683	0.635	**0.690**	**0.690**
MNIST-12345	0.742	0.690	0.720	0.764	**0.921**
COIL20	0.615	0.806	0.635	0.834	**0.861**
Scene	0.641	0.890	0.739	0.871	**0.895**
20NG_rec	0.632	0.904	0.844	0.907	**0.943**
TDT2	0.896	0.989	0.967	**0.998**	**0.998**

4.2 Clustering Evaluation and Parameter Selection

Two standard clustering evaluation metrics Clustering Accuracy (ACC) and Normalized Mutual Information (NMI) are used in our experiments to evaluate the performance of the involved clustering algorithms.

[3] http://people.csail.mit.edu/jrennie/20Newsgroups/
[4] http://www.nist.gov/speech/tests/tdt/tdt98/index.html

Since many clustering algorithms have one or more parameters to be tuned, in order to compare fairly, we run these algorithms under different parameters, and report their best average results. For all clustering algorithms, the number of clusters is set to be equal to the true number of classes for all data sets.

For all graph-based algorithms, such as Normalized Cut, SEC and our NDECSR method, the width parameter σ in Gaussian similarity is automatically selected by self-tuning spectral clustering method [17] and neighborhood size k is tuned from $\{5, 10, 15, 20, 50, 100\}$.

For our NDECSR method, we set $m = \min(\#\mathrm{Dim}, 15)$, where #Dim is the dimensionality of original data. The dimensionality of the desired low dimensional space d is selected from $\{3, 5, 10, 15\}$ and no greater than m for all data sets. The trade-off parameter γ is tuned from $\{10^{-6}, 10^{-3}, 0.01, 0.1, 0.5, 1, 5, 10, 50, 100, 10^{3}\}$.

For SEC, there are two trade-off parameters μ and γ. As the way in SEC [8], we set $\gamma = 1$, and tune μ from $\{10^{-10}, 10^{-7}, 10^{-4}, 10^{-1}, 1, 10^{2}, 10^{5}, 10^{8}\}$.

Table 3. Comparison of Normalized Mutual Information of different methods

	KM	NCut	DKM	SEC	NDECSR
Ionosphere	0.131	0.145	0.142	0.142	**0.358**
Iris	0.716	0.830	0.680	**0.851**	0.844
Wine	0.841	0.875	0.834	0.893	**0.909**
ORL	0.755	0.815	0.788	0.817	**0.823**
MNIST-12345	0.585	0.697	0.510	0.708	**0.824**
COIL20	0.746	0.877	0.765	0.922	**0.943**
Scene	0.508	0.713	0.560	0.721	**0.741**
20NG_rec	0.483	0.724	0.606	0.730	**0.754**
TDT2	0.879	0.961	0.929	**0.992**	0.989

Given the constructed affinity graph and the number of clusters, there is no other parameter in Normalized Cut. DisKmeans needs to determine a regularization parameter γ. We also tune it from $\{10^{-10}, 10^{-7}, 10^{-4}, 10^{-1}, 1, 10^{2}, 10^{5}, 10^{8}\}$.

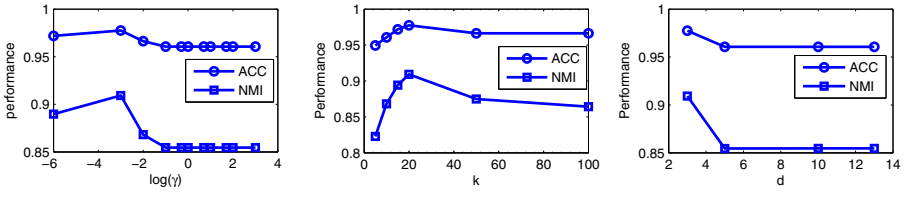
For all methods that produce the continuous assignment matrix, we obtain the final cluster results using the spectral rotation method introduced in [12].

4.3 Clustering Performance

The clustering performance of involved algorithms are compared in Table 2 and Table 3, in both of which the best results are highlighted in boldface. From these comparisons, one can see that our NDECSR method outperforms KM, NCut, DKM and SEC in most cases. Especially on Ionosphere and MNIST_12345 data set, our NDECSR method can significantly improve the clustering performance of the other methods. This demonstrates that learning both smooth low dimensional coordinates and smooth cluster matrix with respect to the data manifold can improve the clustering performance.

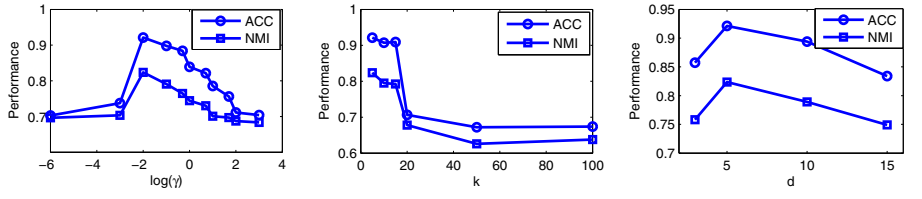
4.4 Clustering Performance vs. Parameters

There are three parameters that should be specified manually in our NDECSR algorithm. We have also performed extensive experiments to evaluate the sensitivity of clustering performance of NDECSR to these parameters. Due to space limitations, we only plot the ACC and NMI versus the parameters on wine and MNIST_12345 data set in Figs. 1, 2, respectively. From these plots, one can see that the clustering performance is a little sensitive to these three parameters. Generally, too large or too small parameters will result in poor performance. Fortunately, for all data sets, we find that under the parameter setting $\gamma = 0.001$, $k = 10$, $d = 10$, our NDECSR method can always obtain good performance. So we can use this parameter setting for simplicity in practice.



(a) Performance vs. γ (b) Performance vs. k (c) Performance vs. d

Fig. 1. Clustering performance of NDECSR with respect to different parameters on Wine data set: (a) Clustering performance vs. γ with $k = 20$, $d = 3$; (b) Clustering performance vs. k with $\gamma = 0.001$, $d = 3$; (c) Clustering performance vs. d with $\gamma = 0.001$, $k = 20$.

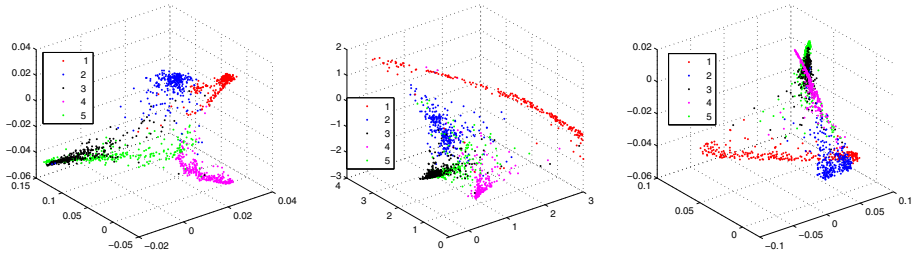


(a) Performance vs. γ (b) Performance vs. k (c) Performance vs. d

Fig. 2. Clustering performance of NDECSR with respect to different parameters on MNIST_12345 data set: (a) Clustering performance vs. γ with $k = 5$, $d = 5$; (b) Clustering performance vs. k with $\gamma = 0.01$, $d = 5$; (c) Clustering performance vs. d with $\gamma = 0.01$, $k = 5$.

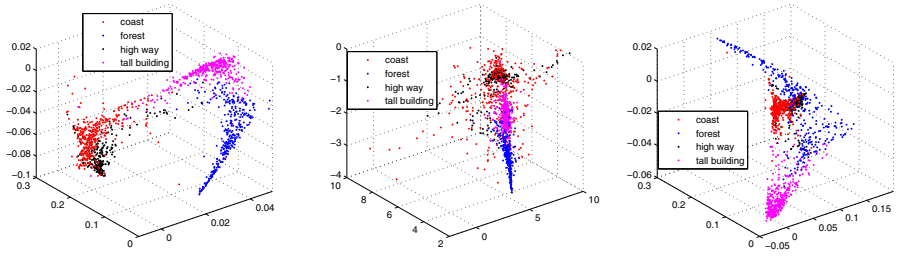
4.5 Comparison on the Embeddings

Besides the clustering result, our algorithm can also yield the nonlinear embedding of original high dimensional data. While the traditional clustering algorithms such as NCut, DKM and SEC have no such an ability. They can either obtain the linear embedding or a fixed dimensionality embedding. Here we compare the 3D-embedding obtained by our method with two popular manifold learning algorithms: LLE [3] and LE [4]. The 3D embeddings of three algorithms on MNIST_12345 and Scene data set are shown in Fig. 3 and 4. As can be seen, our NDECSR method can preserve the cluster structure better than LLE and LE.



(a) 3D embedding of NDECSR (b) 3D embedding of LLE (c) 3D embedding of LE

Fig. 3. Comparison on 3D embeddings of different methods on MNIST_12345 data set: (a) 3D embedding of NDECSR under $\gamma = 0.01$, $k = 5$; (b) 3D embedding of LLE under neighborhood size $k = 10$; (c) 3D embedding of LE under neighborhood size $k = 10$



(a) 3D embedding of NDECSR (b) 3D embedding of LLE (c) 3D embedding of LE

Fig. 4. Comparison on 3D embeddings of different methods on Scene data set: (a) 3D embedding of NDECSR under $\gamma = 0.001$, $k = 5$; (b) 3D embedding of LLE under neighborhood size $k = 10$; (c) 3D embedding of LE under neighborhood size $k = 10$

5 Conclusions

Since traditional dimensionality reduction are developed originally for classification or recovering the geometric structure of data (known as manifold), there exists intrinsic separation between existing DRs and clustering. On the other hand, existing clustering algorithms still have some problems when directly working on high dimensional data. Therefore, in this paper, a novel dimensionality reduction method for clustering called NDECSR is proposed. The key property of the proposed method is that the nonlinear dimensional reduction and clustering task can be achieved simultaneously in a unified framework. To our best knowledge, this is the first method that can yield both nonlinear embedding and cluster result in a unified framework in the literature. Extensive experiments on three data sets from UCI machine learning repository and real world image and text document data sets demonstrate its effectiveness as a cluster algorithm as well as its effectiveness as a nonlinear dimensionality reduction method.

References

1. Jolliffe, I.: Principal component analysis. Springer, Heidelberg (2002)
2. Tenenbaum, J., Silva, V., Langford, J.: A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319–2323 (2000)
3. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326 (2000)
4. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation 15(6), 1373–1396 (2003)
5. Duda, R., Hart, P., Stork, D.: Pattern classification. John Wiley & Sons, Chichester (2001)
6. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. 17, 395–416 (2007)
7. Yan, S., Xu, D., Zhang, B., Zhang, H.J., Yang, Q., Lin, S.: Graph embedding and extensions: A general framework for dimensionality reduction. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(1), 40–51 (2007)
8. Nie, F., Xu, D., Tsang, I.W., Zhang, C.: Spectral embedded clustering. In: Proceedings of the 21st International Jont Conference on Artifical Intelligence, pp. 1181–1186. Morgan Kaufmann Publishers Inc., San Francisco (2009)
9. Belkin, M., Niyogi, P.: Semi-supervised learning on riemannian manifolds. Machine Learning 56(1), 209–239 (2004)
10. Li, Z., Liu, J., Tang, X.: Constrained clustering via spectral regularization. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009. IEEE, Los Alamitos (2009)
11. Zha, H., He, X., Ding, C., Simon, H.: Spectral relaxation for k-means clustering. In: Advances in Neural Information Processing Systems, vol. 2, pp. 1057–1064. MIT Press, Cambridge (2002)
12. Jianbo, S.Y., Yu, S.X., Shi, J.: Multiclass spectral clustering. In: Proceedings of 9th IEEE International Conference on Computer Vision 2003, pp. 313–319 (2003)
13. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
14. Ye, J., Zhao, Z., Wu, M.: Discriminative k-means for clustering. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems, vol. 20, pp. 1649–1656. MIT Press, Cambridge (2008)
15. Nene, S.A., Nayar, S.K., Murase, H.: Columbia object image library (coil-20). Technical report, Dept. Comput. Sci., Columbia Univ., New York (1996)
16. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. International Journal of Computer Vision 42(3), 145–175 (2001)
17. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 17, pp. 1601–1608. MIT Press, Cambridge (2005)

An Adaptive Fuzzy k-Nearest Neighbor Method Based on Parallel Particle Swarm Optimization for Bankruptcy Prediction

Hui-Ling Chen, Da-You Liu*, Bo Yang, Jie Liu, Gang Wang, and Su-Jing Wang

Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of EducationJilin University, Changchun 130012, China
chenhuiling.jlu@gmail.com, {liudy,ybo,liu_jie}@jlu.edu.cn,
wanggang.jlu@gmail.com, wangsj08@mails.jlu.edu.cn

Abstract. This study proposes an efficient non-parametric classifier for bankruptcy prediction using an adaptive fuzzy k-nearest neighbor (FKNN) method, where the nearest neighbor k and the fuzzy strength parameter m are adaptively specified by the particle swarm optimization (PSO) approach. In addition to performing the parameter optimization for FKNN, PSO is utilized to choose the most discriminative subset of features for prediction as well. Time varying acceleration coefficients (TVAC) and inertia weight (TVIW) are employed to efficiently control the local and global search ability of PSO. Moreover, both the continuous and binary PSO are implemented in parallel on a multi-core platform. The resultant bankruptcy prediction model, named PTVPSO-FKNN, is compared with three classification methods on a real-world case. The obtained results clearly confirm the superiority of the developed model as compared to the other three methods in terms of Classification accuracy, Type I error, Type II error and AUC (area under the receiver operating characteristic (ROC) curve) criterion. It is also observed that the PTVPSO-FKNN is a powerful feature selection tool which has indentified a subset of best discriminative features. Additionally, the proposed model has gained a great deal of efficiency in terms of CPU time owing to the parallel implementation.

Keywords: Fuzzy k-nearest neighbor, Parallel computing, Particle swarm optimization, Feature selection, Bankruptcy prediction.

1 Introduction

Accurately identifying the potentially financial failure of companies remains a goal of many stakeholders involved. Because there is no underlying economic theory of bankruptcy, searching for more accurate bankruptcy prediction models remains the goal in the field of the bankruptcy prediction. A fair amount of models has been developed for bankruptcy prediction. These models have progressed from statistical methods to the artificial intelligence (AI) approach.

* Corresponding author.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 249–264, 2011.

A number of statistical methods such as the simple univariate analysis, multivariate discriminant analysis technique, logistic regression approach and factor analysis technique have been typically used for financial applications including bankruptcy prediction. Recent studies in the AI approach, such as artificial neural networks (ANN) , rough set theory , support vector machines (SVM) , k-nearest neighbor method (KNN) and Bayesian network models have also been successfully applied to bankruptcy prediction (see [1] [2]). Among these techniques, ANN has become one of the most popular techniques for the prediction of corporate bankruptcy due to its high prediction accuracy. However, a major disadvantage of ANN lies in their knowledge representation. The black box nature of ANN makes it difficult for humans to understand how the networks predict the bankruptcy.

Compared with ANN, KNN is simple, easily interpretable and can achieve acceptable accuracy rate. Albeit these advantages, the standard KNN methods place equal weights on all the selected neighbors regardless of their distances from the query point. In this way, once the class has been assigned, there is no indication of the significance of membership to indicate how much the instance belongs to a particular class. An improvement over the standard KNN classifier is the Fuzzy k-nearest neighbor classifier (FKNN) [3], which uses concepts from fuzzy logic to assign degree of membership to different classes while considering the distance of its k nearest neighbors. The FKNN method has been frequently used for the classification of biological data, image data and so on. Nevertheless, only few works have paid attention to using FKNN to classify the financial data. Bian et al. [4] used FKNN as a reference classifier in their experiments in order to show the superiority of the proposed Fuzzy-rough KNN method, which incorporated the rough set theory into FKNN to further improve the accuracy of bankruptcy prediction. However, they did not comprehensively investigate the nearest neighbors k and the fuzzy strength parameter m, which play a significant role in improving the prediction power for FKNN. This study aims to explore the full potential of FKNN by automatically determining k and m to exploit the maximum classification accuracy for bankruptcy prediction.

Besides choosing a good learning algorithm, feature selection is also an important issue in building the bankruptcy prediction models [5], which refers to choosing subset of attributes from the set of original attributes. The purpose of the feature selection is to identify the significant features, eliminate the irrelevant of dispensable features and build a good learning model. In bankruptcy prediction, genetic algorithms (GA) are usually used to select a subset of input features or to find appropriate hyper-parameter values of a predictor. Compared with GA, particle swarm optimization (PSO) algorithm has no crossover and mutation operators, it is simple and computationally inexpensive both in memory and runtime. In this work, we will focus on exploring the PSO-based parameter optimization and feature selection approach. The continuous PSO algorithm will be employed to evolve an adaptive FKNN, where the nearest neighbor k and the

fuzzy strength parameter m are adaptively specified. On the other hand, the binary PSO will be used as a feature selection vehicle to identify the most informative features as well.

When dealing with the practical problems, the evolutionary-based methods such as the PSO and GA will cost a lot of computational time. There is an urgent need to improve the performance using high-performance computing techniques. For this reason, it is one of the major purposes of this paper to use a parallel environment to speed up the search and optimization process. Both the continuous and binary time variant PSO are implemented on a multi-core platform using OpenMP (Open Multi-Processing) which is a portable, scalable model that gives programmers a simple and flexible interface for developing parallel applications for platforms [6]. The efficacy of the proposed bankruptcy prediction model PTVPSO-FKNN is compared with three reference classification methods on a real-world case. All these classifiers are compared with respect to the classification accuracy, Type I error, Type II error and the AUC (area under the receiver operating characteristic (ROC) curve) criterion. The experimental results demonstrate that the proposed model can not only obtain the most appropriate parameters but also show high discriminating power as a feature selection tool. Further comparison is also made between the parallel model and serial model. Based on the experiments conducted, it is inferred that the parallel model PTVPSO-FKNN can significantly reduce the computational time.

The rest of the paper is organized as follows. In Section 2, we give a brief description of the fuzzy k-nearest neighbor method (FKNN) and particle swarm optimization algorithm (PSO). Section 3 proposes our model, the simultaneous optimization of relevant parameters and feature subset by the PSO approach in a parallel environment. In the next section, the detailed experimental design is presented, and Section 5 describes all the empirical results and discussion. Finally, Conclusions are summarized in Section 6.

2 Background Materials

2.1 Fuzzy k-Nearest Neighbor Algorithm (FKNN)

The k-nearest neighbor algorithm (KNN) is one of the oldest and simplest non parametric pattern classification methods [7]. In the KNN algorithm a class is assigned according to the most common class amongst its k nearest neighbors. In 1985, Keller proposed a fuzzy version of KNN by incorporating the fuzzy set theory into the KNN algorithm, and named it as "fuzzy KNN classifier algorithm" (FKNN) [3]. According to his approach, rather than individual classes as in KNN, the fuzzy memberships of samples are assigned to different categories according to the following formulation:

$$u_i(x) = \frac{\sum_{j=1}^{k} u_{ij}(1/||x - x_j||^{2/(m-1)})}{\sum_{j=1}^{k} (1/||x - x_j||^{2/(m-1)})} \tag{1}$$

where $i = 1, 2, \cdots, c$, and $j = 1, 2, \cdots, k$, with c number of classes and k number of nearest neighbors. The fuzzy strength parameter m is used to determine how heavily the distance is weighted when calculating each neighbor's contribution to the membership value, and its value is usually chosen as $m \in (1, +\infty)$. $\|x - x_j\|$ is the Euclidean distance between x and its jth nearest neighbor x_j. And u_{ij} is the membership degree of the pattern x_j from the training set to the class i, among the k nearest neighbors of x. There are two ways to define u_{ij}, one way is the crisp membership, i.e., each training pattern has complete membership in their known class and non-memberships in all other classes. The other way is the constrained fuzzy membership, i.e., the k nearest neighbors of each training pattern (say x_k) are found, and the membership of x_k in each class is assigned as:

$$u_{ij}(x_k) = \begin{cases} 0.51 + (n_j/K) * 0.49, & \text{if } j = i \\ (n_j/K) * 0.49, & \text{otherwise.} \end{cases} \tag{2}$$

The value n_j is the number of neighbors found which belong to the jth class. In our experiments, we have found that the second way lead to better classification accuracy. After calculating all the memberships for a query sample, it is assigned to the class with which it has the highest membership value.

2.2 Time Variant Particle Swarm Optimization (TVPSO)

Particle swarm optimization (PSO) was first developed by Kennedy and Eberhart [8]. In PSO each individual is treated as a particle in d-dimensional space, and each particle has a position and velocity. The position vector of the ith particle is represented as $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$, and its corresponding velocity is represented as $V_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,d})$. The velocity and position are updated as follows:

$$v_{i,j}^{n+1} = w \times v_{i,j}^{n} + c_1 \times r_1(p_{i,j}^{n} - x_{i,j}^{n}) + c_2 \times r_2(p_{g,j}^{n} - x_{i,j}^{n}) \tag{3}$$

$$x_{i,j}^{n+1} = x_{i,j}^{n} + v_{i,j}^{n+1}, j \ = 1, 2, \cdots, d \tag{4}$$

where vector $P_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,d})$ represents the best previous position of the ith particle that gives the best fitness value, which is known as the personal best position *(pbest)*. Vector $P_g = (p_{g,1}, p_{g,2}, \ldots, p_{g,d})$ is the best particle among all the particles in the population, which is known as the global best position *(gbest)*. r_1 and r_2 are random numbers, generated uniformly in the range $[0, 1]$. The velocity $v_{i,j}$ is restricted to the range $[-v_{max}, v_{max}]$. Inertia weight w is updated according to the following equation:

$$w = w_{\min} + (w_{\max} - w_{\min}) \frac{(t_{\max} - t)}{t_{\max}} \tag{5}$$

where w_{max}, w_{min} are the predefined maximum and minimum values of the inertia weight w, t is the current iteration of the algorithm and t_{max} is the maximum number of iterations. Eq. (5) is also known as Time varying inertia weight

(TVIW), which will be incorporated to the TVPSO. c_1 and c_2 are acceleration coefficients, to better balance the search space between the global exploration and local exploitation, Time varying acceleration coefficients (TVAC) have been introduced in [9]. This concept will be adopted in this study to ensure the better search for the solutions. The core idea of TVAC is that c_1 decreases from its initial value of c_{1i} to c_{1f}, while c_2 increases from c_{2i} to c_{2f} using the following equations as in [9]. TVAC can be mathematically represented as follows:

$$c_1 = (c_{1f} - c_{1i})\frac{t}{t_{\max}} + c_{1i} \tag{6}$$

$$c_2 = (c_{2f} - c_{2i})\frac{t}{t_{\max}} + c_{2i} \tag{7}$$

where c_{1f}, c_{1i}, c_{2f} and c_{2i} are constants, t is the current iteration of the algorithm and t_{max} is the maximum number of iterations. For the binary PSO, one discrete PSO version introduced by Kennedy and Eberhart [10] was employed to act as the feature selection tool. In the binary PSO, A sigmoid function is applied to transform the velocity from continuous space to probability space:

$$sig(v_{i,j}) = \frac{1}{1 + \exp(-v_{i,j})}, j \ = 1, 2, \ldots, d \tag{8}$$

The velocity update Eq. (3) keeps unchanged except that $x_{i,j}$, $p_{i,j}$ and $p_{g,j} \in \{0, 1\}$, and in order to ensure that bit can transfer between 1 and 0 with a positive probability, v_{max} was introduced to limit $v_{i,j}$. The new particle position is updated using the following rule:

$$x_{ij}^{n+1} = \begin{cases} 1, & \text{if } rnd < sig(v_{i,j}) \\ 0, & \text{if } rnd \geq sig(v_{i,j}) \end{cases}, j \ = 1, 2, \cdots, d \tag{9}$$

where $sig(v_{i,j})$ is calculated according to Eq. (8), and rand is a uniform random number in the range $[0, 1]$.

3 Proposed PTVPSO-FKNN Prediction Model

In this section, we describe the proposed PTVPSO-FKNN model for bankruptcy prediction. As mentioned in the Introduction, the aim of this model is to optimize the FKNN classifier by automatically: 1) determining the nearest neighbor k and the fuzzy strength parameter m and 2) identifying the subset of best discriminative features. In order to achieve this goal, the continuous and binary time variant PSO are combined together to dynamically conduct the parameter optimization and feature selection. The obtained appropriate feature subset can served as the input into the FKNN classifier to conduct the classification task. Here, we first describe the model based on the serial PSO algorithm, termed TVPSO-FKNN, and then implement it in parallel.

3.1 TVPSO-FKNN Model Based on the Serial PSO Algorithm

The flowchart of the TVPSO-FKNN model for bankruptcy prediction was constructed through the following main steps as shown in Fig. 1.

- Step 1: Encode the particle with n+2 dimensions. The first two dimensions are k and m which are continuous values. The remaining n dimensions is Boolean features mask, which is represented by discrete value, '1' indicates the feature is selected, and '0' represents the feature is discarded.
- Step 2: Initialize the individuals of the population with random numbers. Meanwhile, specify the PSO parameters including the lower and upper bounds of the velocity, the size of particles, the number of iterations, etc.
- Step 3: Train the FKNN with the selected feature vector in Step 2.
- Step 4: It is well known that higher the AUC value the better the classifier is said to be. And the particle with high AUC value and the small number of selected features can produce a high fitness value. Hence, we took both of them into consideration in designing the fitness function, The fitness value was calculated according to the following objective function:

$$\begin{cases} f_1 = \text{AUC} \\ f_2 = (1 - \frac{\sum_{j=1}^{n} f_{ti}}{n}) \\ f = \alpha \times f_1 + \beta \times f_2 \end{cases} \tag{10}$$

  where variable AUC in the first sub-objective function f_1 represents the area under the ROC curve achieved by the FKNN classifier via K-fold cross-validation (CV), here K=5. Note that here the 5-fold CV is used to determine the optimal parameters (including k and m) which is different from the outer loop of 10-fold CV, which is used to do the performance estimation. In the second sub-objective function f_2, f_{ti} is the value of feature mask ('1' represents that feature is selected and '0' indicates that feature is discarded), n is the total number of features. The weighted summation of the two sub-objective functions is selected as the final sub-objective function. In the function f, variable α is the weight for FKNN classification accuracy, β indicates the weight for the selected features. The weight can be adjusted to a proper value depends on the importance of the sub-objective function. Because the classification performance more depend on the classification accuracy, hence the α value is set as much bigger than that of β. According to our preliminary experiments, the value of α and β were taken as 0.85 and 0.15 respectively. After the fitness value was obtained, the global optimal fitness was saved as *gfit*, personal optimal fitness as *pfit*, global optimal particle as *gbest* and personal optimal particle as *pbest*.
- Step 5: Increase the number of iteration.
- Step 6: Increase the number of population. Update the position and velocity of k, m using Eqs.(3-4) and the features using Eq.(3), Eqs.(8-9) in each particle.
- Step 7: Train the FKNN classifier with the feature vector obtained in Step 6 and calculate the fitness value of each particle according to Eq. (10). Notice

that PSO is used for optimization tasks where the nearest neighbor k to be optimized is integer number. Hence, an extra step is taken to round the encoded value k to the nearest integer number before the particle is evaluated.

- Step 8: Update the personal optimal fitness (*pfit*) and personal optimal position (*pbest*) by comparing the current fitness value with the *pfit* stored in the memory. If the current fitness is dominated by the *pfit* stored in the memory, then keep the *pfit* and *pbest* in the memory; otherwise, replace the *pfit* and *pbest* in the memory with the current fitness value and particle position.
- Step 9: If the size of the population is reached, then go to Step 10. Otherwise, go to Step 6.
- Step 10: Update the global optimal fitness (*gfit*) and global optimal particle (*gbest*) by comparing the *gfit* with the optimal *pfit* from the whole population, If the current optimal *pfit* is dominated by the *gfit* stored in the memory, then keep the *gfit* and *gbest* in the memory; otherwise, replace the *gfit* and *gbest* in the memory with the current optimal *pfit* and the optimal *pbest* from the whole population.
- Step 11: If the stopping criteria are satisfied, then go to Step 12. Otherwise, go to Step 5. The termination criteria are that the iteration number reaches the maximum number of iterations or the value of *gfit* does not improve after 100 consecutive iterations.
- Step 12: Get the optimal (k, m) and feature subset from the best particle (*gbest*).

3.2 Parallel Implementation of the TVPSO-FKNN Model on the Multi-core Platform (PTVPSO-FKNN)

In this section, we put forward a parallel implementation of TVPSO-FKNN model which is performed on multi-core processor by using OpenMP. The architecture of the multi-core platform is divided into three lays as shown in Fig. 2(a): 1) TVPSO-FKNN: It consists of a number of particles, which can supply computing requirements. The parallel algorithm controls the iterations of particles and each particle is calculated separately. 2) OpenMP: It guarantees to implement parallel synchronization and establish the communications with operating system (OS). The main part of OpenMP is scheduler, which provides the system with job scheduling and allocation. 3) Multi-core processor: The job is dispatched by OpenMP via OS.

The pseudo-code of the parallel TVPSO-FKNN is summarized in Algorithm 1.

4 Experimental Design

4.1 Data Description

The financial data used for this study was taken from Wieslaw [11] dataset which contains 30 financial ratios and 240 cases in total (112 from bankrupt

Algorithm 1. PTVPSOFKNN

```
Initialize system parameters.
Train FKNN model.
//current number of iteration (cni), maximum number of iteration (mni)
while cni < mni do
  for each particle do
    Update position.
    Update velocity.
    Train FKNN model.
    Calculate fitness.
    Calculate pfit. // personal optimal fitness (pfit)
    Calculate pbest. // personal optimal position (pbest)
  end for
  Calculate gfit. // global optimal fitness (gfit)
  Calculate gbest. // global optimal particle (gbest)
  cni =cni + 1.
end while
```

Polish companies and 128 from non-bankrupt ones between 1997 and 2001). All the observations cover the period spanning 2 to 5 years before bankruptcy toke place. It should be noted that the size of the data set is not that large compared to the majority of bankruptcy prediction studies. However, according to [12], the dataset is reliable since increasing the dataset length does not lead to the accuracy increase. Fig. 2(b) illustrates the distribution of the two classes of 240 samples in the subspace formed by the two best features according to the principal component analysis (PCA) algorithm. As shown in this figure, there is apparently strong overlap between the bankrupt companies and non-bankrupt ones.

Data was normalized by scaling them into the interval of $[-1, 1]$. In order to gain an unbiased estimate of the generalization accuracy, the k-fold CV presented by Salzberg [13] was used to evaluate the classification accuracy. This study set k as 10, i.e., the data was divided into ten subsets. Each time, one of the 10 subsets is used as the test set and the other 9 subsets are put together to form a training set. Then the average error across all 10 trials is computed. The advantage of this method is that all of the test sets are independent and the reliability of the results could be improved.

4.2 Experimental Setup

The proposed PTVPSO-FKNN model was implemented using Visual C++ 2008 and OpenMP. For SVM, LIBSVM implementation is utilized, which was originally developed by Chang and Lin [14]. We implemented the PSO algorithm, FKNN and KNN from scratch. The MLP was created, trained and implemented using Matlab neural network toolbox with BP and the training algorithm of Levenberg-Marquardt. The computer is Intel Quad-Core Xeon 2.0 GHz CPU; 4 GB RAM and the system is Windows Server 2003.

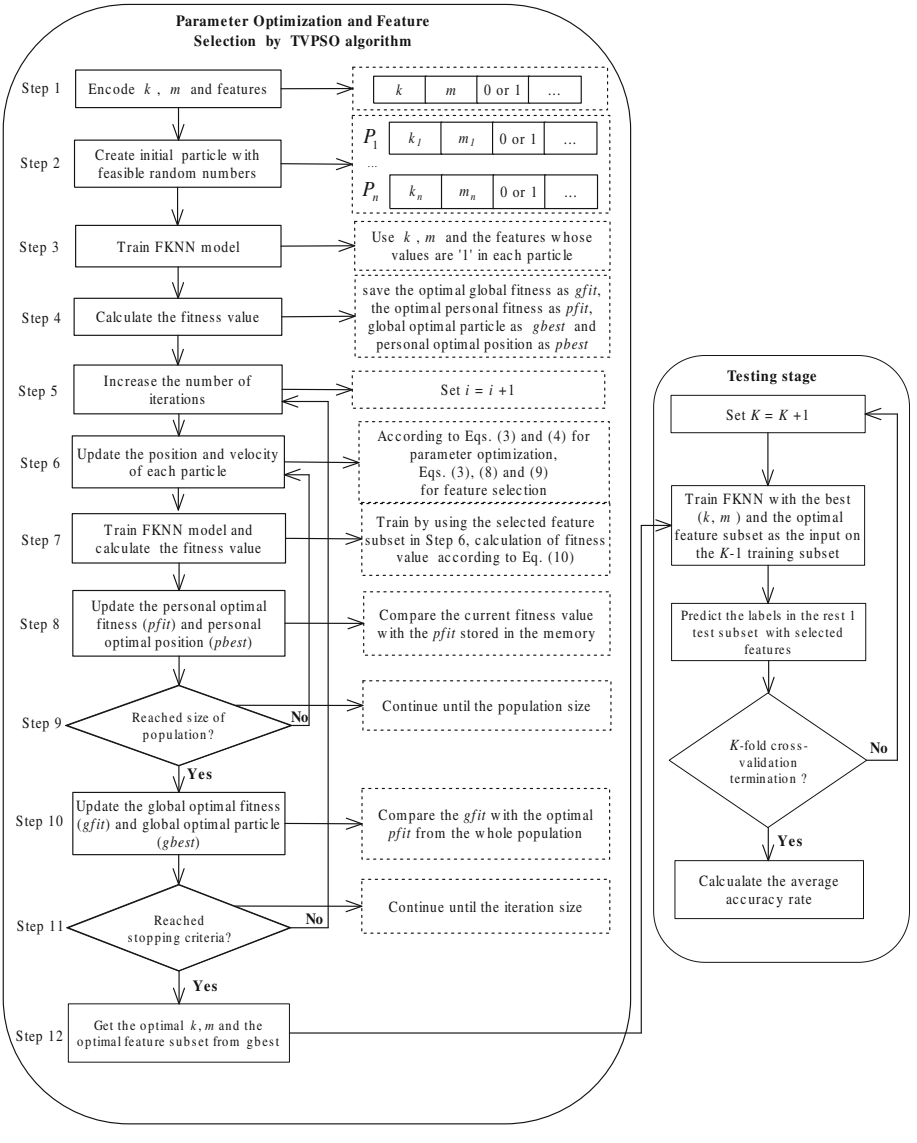
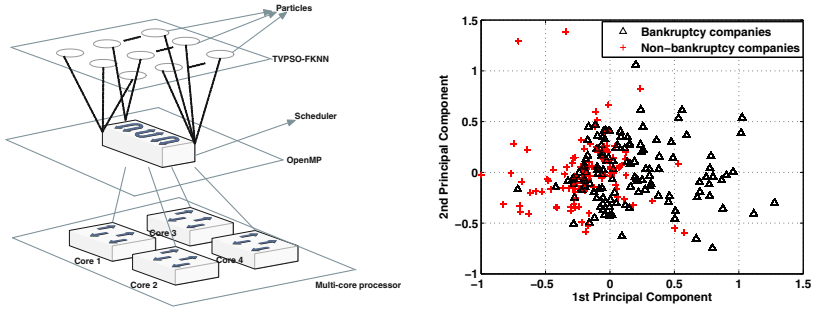


Fig. 1. Overall procedure of the TVPSO-FKNN model



(a) (b)

Fig. 2. (a)The architecture of parallel running environment of TVPSO-FKNN. (b)Two-dimensional distribution of the two classes (bankrupt and non-bankrupt) in the subspace formed by the best couple of features obtained with the PCA algorithm.

The detail parameter setting for PTVPSO-FKNN was set as follows. The number of the iterations and particles was set to 250 and 8, respectively. The searching ranges for k and m are as follows: $k \in [1, 100]$ and $m \in [1, 10]$. v_{max} was set about 60% of the dynamic range of the variable on each dimension for the continuous type of dimensions. Therefore, $[-v_{max}, v_{max}]$ was predefined as [0.6, 60] for parameter k, and as $[0.6, 6]$ for parameter m. For the discrete type particle for feature selection, $[-v_{max}, v_{max}]$ was set as $[-6, 6]$. As suggested in [9],c_{1i},c_{1f},c_{2i} and c_{2f} were set as follows: $c_{1i} = 2.5$,$c_{1f} = 0.5$,$c_{2i} = 0.5$,$c_{2f} = 2.5$. According to our preliminary experiment, w_{max} and w_{min} were set to 0.9 and 0.4, respectively.

For SVM, we considered the nonlinear SVM based on the popular Gaussian (RBF) kernel, and a grid-search technique [15] was employed using 10-fold CV to find out the optimal parameter values of RBF kernel function. The range of the related parameters C and γ were varied between $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ and $\gamma = \{2^{-15}, 2^{-13}, \ldots, 2^{1}\}$. For KNN, we found the best result was achieved when $k = 1$ by using 10-fold CV. Therefore, we selected $k = 1$ for the subsequent analysis. Concerning MLP, we used the three layer back-propagation network to train ANN. We tried different settings of the number of nodes in the hidden layers (5, 10, 15, 20, 25 and 30) and the different learning epochs (50, 100, 200 and 300) as the stopping criteria for training. The best result was obtained with the hidden layer of 15 and the learning epoch of 200.

4.3 Measure for Performance Evaluation

Type I error, Type II error, total classification accuracy (ACC) and the area under the Receiver Operating Characteristic curve (AUC) [16] were used to test the performance of the proposed PTVPSO-FKNN model. They were the most widely used measures to assess the performance of bankruptcy prediction systems [1]. Type I and Type II errors were two important measures which

described how well the classifier discriminates between case with non-bankruptcy and with bankruptcy. Type I error measures the proportion of bankrupt cases which are incorrectly identified as non-bankrupt ones. Type II error measures the proportion of non-bankrupt cases which are incorrectly identified as bankrupt ones. The receiver operating characteristic (ROC) curve is a graphical display that gives the measure of the predictive accuracy of a logistic model [16]. The curve displays the true positive rate and false positive rate. AUC is the area under the ROC curve, which is one of the best methods for comparing classifiers in two-class problems.

5 Experimental Results and Discussion

5.1 Experiment I: Classification in the Whole Original Feature Space

As mentioned earlier, in this experiment we evaluated the effectiveness of the proposed model on the entire feature space with 30 features (financial ratios). In order to verify the effectiveness of the proposed model, TVPSO-FKNN was compared with three other reference classifiers (SVM, KNN and ANN). Table 1 shows the results achieved with all four investigated classifiers (PTVPSO-FKNN, SVM, KNN and ANN) for the financial data with the form of 'average ± standard deviation'. It is well known that higher the AUC value the better the classifier is said to be. Accordingly, the classifiers are arranged in the descending order of AUC in the table. As clearly indicated in the table, PTVPSO-FKNN outperforms all other methods with the classification accuracy of 81.67%, Type I error of 17.58%, Type II error of 19.04% and AUC of 81.69%. MLP is next to PTVPSO-FKNN with classification accuracy of 77.92%, Type I error of 20.84%, Type II error of 21.46% and AUC of 78.71%, followed by KNN and SVM. The superiority of the PTVPSO-FKNN is statistically significant as shown by the paired t-test in Tables (2-3), where the significant level is 5%. The results are interesting and exciting, it suggests that the FKNN approach can become a promising alternative bankruptcy prediction tool in financial decision-making, where SVM and ANN are known to be the best models [2].

The better performance of the proposed model can be explained by the fact that the TVPSO has aided the FKNN classifier to achieve the maximum classification performance by automatically detecting the optimal nearest neighbor k

Table 1. The ACC, Type I and Type II errors and AUC achieved with different classifiers

Classifiers	ACC (%)	Type I error (%)	Type II error (%)	AUC (%)
PTVPSO-FKNN	81.67±2.15	17.58±0.78	19.04±3.96	81.69±2.04
SVM	76.67±4.65	18.96±8.46	26.55±7.93	77.26±5.62
KNN	78.75±3.65	21.46±5.07	21.39±4.13	78.57±3.78
MLP	77.92±5.22	20.84±7.21	21.46±9.84	78.71±6.48

Table 2. Paired t-test results of Type I and Type II error

Type I error / Type II error	TVPSO-FKNN t-value (significance)
MLP	-2.243(0.037)/-2.589(0.047)
KNN	-2.332(0.045)/-2.366(0.042)
SVM	-3.045(0.026)/-3.122(0.032)

Negative values indicate that the ith classifier has a Type I error /Type II error higher than that of the jth one.

Table 3. Paired t-test results of ACC and AUC

ACC / AUC	TVPSO-FKNN t-value (significance)
MLP	-3.345(0.017)/-3.623(0.021)
KNN	-3.280(0.009)/-3.168(0.011)
SVM	-4.458(0.005)/-4.854(0.023)

Negative values indicate that the ith classifier has an ACC/AUC lower than that of the jth one.

Table 4. The detailed parameter values obtained through 10-fold CV

Fold	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
k	23	33	55	14	1	25	86	43	22	15
m	1.27	1.33	1.39	1.35	1.29	1.34	1.67	1.45	1.32	3.00

and the fuzzy strength parameter m. The detailed values of parameters k and m via 10-fold CV using the proposed model is shown in Table 4. From the table, it can be observed that the values of k and m are different for each fold of the data. And according to our preliminary experiment, they can be varied automatically when perform another run of 10-fold CV. The explanation lies in the fact that the two parameters are evolved together by the TVPSO algorithm according to the specific distribution of the training data at hand. It indicates that the optimal values of k and m can always be adaptively specified by TVPSO during each run of the experiment. Moreover, it is interesting to see that the standard deviation for the acquired performance by the PTVPSO-FKNN is much smaller than that of the other three classifiers, which indicates consistency and stability of the proposed model.

5.2 Experiment II: Classification Using the PTVPSO-FKNN Model with Feature Selection

As described earlier, the proposed PTVPSO-FKNN model aimed at enhancing the FKNN classification process by not only dealing with the parameters optimization but also automatically identifying the subset of the most discriminative features. In this experiment, we attempt to explore the capability of the

PTVPSO-FKNN to further boost the performance of the FKNN classifier by using the TVPSO. Table 5 lists the best results of PTVPSO-FKNN with and without feature selection for Wieslaw dataset. As shown in this table, results obtained using PTVPSO-FKNN with feature selection significantly outperforms PTVPSO-FKNN without feature selection in terms of the Type I error, Type II error, AUC and classification accuracy at the statistical significance level of 5%. By using feature selection, the classification accuracy, AUC values, Type I error and Type II error have been improved by 2.5%, 2.55%, 1.71% and 3.38% on average, respectively.

Table 5. Experimental results of the PTVPSO-FKNN with and without feature selection(%)

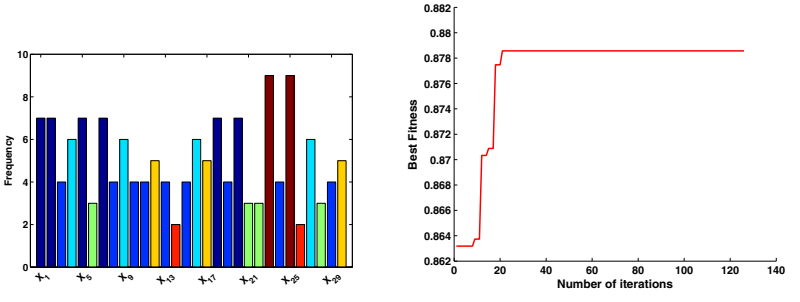
Performance metric	PTVPSO-FKNN without feature selection	PTVPSO-FKNN with feature selection	Paired $t-$test p-value
Type I error	17.58±0.78	15.87±2.42	0.0429
Type II error	19.04±3.96	15.66±1.94	0.0202
AUC	81.69±2.04	84.24±1.75	0.0029
ACC	81.67±2.15	84.17±1.76	0.0051

To explore how many features and what features are selected during the PSO feature selection procedure, we further conducted an experiment on the Wieslaw dataset to investigate the detail of the feature selection mechanism of the PSO algorithm. The original numbers of features of the dataset is 30. As shown in Table 6, not all features are selected for classification after the feature selection. Furthermore, feature selection has increased the classification accuracy, as demonstrated in Table 5. The average number of selected features by PTVPSO-FKNN is 15.3, and its most important features are X_1, X_2, X_4, X_5, X_7, X_9, X_{16}, X_{18}, X_{20}, X_{23}, X_{25} and X_{27}, which can be found in the frequency of the selected features of 10-fold CV as shown in Fig. 3(a). It should be noticed that important features (financial ratios) selected by the proposed model are indeed important from the knowledge perspective also as they are related to current liabilities and long term liabilities, current assets, shareholders' equity and cash, sales, inventory, working capital, net profit, receivables, liabilities, total assets.

To observe the evolutionary process in PTVPSO-FKNN, Fig. 3(b) shows the evolution of the best fitness for fold #1 during 10-fold CV. The evolutionary processes are quite interesting. It can be observed that the fitness curves gradually improved from iteration 1 to 130 and exhibited no significant improvements after iteration 22, eventually stopped at the iteration 130 where the particles reached the stopping criterion(100 successively same *gbest* values). The fitness increase rapidly in the beginning of the evolution, after certain number of generations, it starts increasing slowly. During the latter part of the evolution, the fitness keeps stability until the stopping criterion is satisfied. It demonstrates that PTVPSO-FKNN can converge quickly toward the global optima, and fine tune the solutions very efficiently. The phenomenon illustrates the effectiveness of PTVPSO-FKNN in simultaneously evolving the parameters (k and m) and the features through using TVPSO algorithm.

Table 6. The subset of features selected by PTVPSO-FKNN via 10-fold CV

Fold	Selected features
#1	X_2 X_4 X_5 X_7 X_{10} X_{11} X_{12} X_{15} X_{20} X_{22} X_{23} X_{26} X_{27}
#2	X_1 X_3 X_4 X_6 X_7 X_8 X_{11} X_{13} X_{15} X_{16} X_{17} X_{18} X_{19} X_{20} X_{23} X_{25} X_{30}
#3	X_1 X_2 X_4 X_6 X_7 X_9 X_{13} X_{16} X_{20} X_{22} X_{23} X_{24} X_{25} X_{27}
#4	X_1 X_2 X_3 X_4 X_5 X_9 X_{10} X_{12} X_{13} X_{15} X_{17} X_{18} X_{20} X_{22} X_{23} X_{24} X_{25} X_{29}
#5	X_1 X_2 X_3 X_6 X_7 X_8 X_9 X_{10} X_{11} X_{12} X_{15} X_{18} X_{19} X_{20} X_{23} X_{25} X_{27} X_{28} X_{29} X_{30}
#6	X_5 X_7 X_9 X_{14} X_{17} X_{18} X_{19} X_{21} X_{23} X_{24} X_{25} X_{27} X_{30}
#7	X_2X_4 X_5 X_7 X_8 X_{12} X_{13} X_{16} X_{17} X_{18} X_{21} X_{23} X_{25} X_{29} X_{30}
#8	X_1 X_2 X_3 X_4 X_5 X_7 X_8 X_{16} X_{19} X_{20} X_{25} X_{27} X_{29}
#9	X_1 X_5 X_9 X_{12} X_{16} X_{18} X_{20} X_{23} X_{24} X_{25} X_{26} X_{28}
#10	X_1 X_2 X_5 X_8 X_9 X_{10} X_{11} X_{14} X_{15} X_{16} X_{17} X_{18} X_{21} X_{23} X_{25} X_{27} X_{28} X_{30}



(a) (b)

Fig. 3. (a) The frequency of the selected features in 10-fold CV on Wieslaw dataset. (b) The best fitness during the training stage for fold #1.

5.3 Experiment III: Comparison between the Parallel TVPSO-FKNN Model and the Serial One

In order to reduce further the running time of the serial TVPSO-FKNN model, we implemented the TVPSO-FKNN model on a multi-core platform. To validate the efficiency of the parallel version, here we attempted to compare the performance of the PTVPSO-FKNN with that of TVPSO-FKNN. Table 7 reported the best results of Type I error, Type II error, ACC, AUC and the average computational time in seconds using the two models. It can be seen that PTVPSO-FKNN and TVPSO-FKNN give almost the same results, the minor different results between two models may be attributed to different partitions of the data are chosen when perform different runs of 10-fold CV. Thus, it verifies the correctness of the parallel design and implementation. However, the training time for the TVPSO-FKNN was 3.3 times that of the PTVPSO-FKNN, which

indicates that the TVPSO-FKNN has benefited a great deal from the parallel implementation with respect to the computational time. Additionally, it should be noted that only a quad-core processor was used in this experiment, thus the computational time will be further reduced with increase of the cores.

Table 7. The performance comparison of PTVPSO-FKNN with TVPSO-FKNN

Performance metric	PTVPSO-FKNN	TVPSO-FKNN
Type I error (%)	15.87±2.42	15.53±2.56
Type II error (%)	15.66±1.94	15.95±1.87
AUC (%)	84.24±1.75	84.26±1.98
ACC (%)	84.17±1.76	84.20±1.55
CPU Time (s)	1150.46±23.34	3796.51±30.45

6 Conclusions

This study provides an attractive model PTVPSO-FKNN for bankruptcy prediction. The main novelty of this model is in the proposed TVPSO-based approach, which aims at aiding the FKNN classifier to achieve the maximum classification performance. On the one hand, the continuous TVPSO is employed to adaptively specify the two important parameters k and m of the FKNN classifier. On the other hand, the binary TVPSO is adopted to identify the most discriminative features. Moreover, both the continuous and binary TVPSO are implemented in a parallel environment to reduce further the computational time. The experimental results demonstrate that the developed model performs significantly better than the other three state-of-the-art classifiers (KNN, SVM and MLP) in financial application field in terms of the Type I error, Type II error, ACC and AUC on a real life dataset. Moreover, the experiment reveals that the PTVPSO-FKNN is also a powerful feature selection tool which has detected a subset of best discriminative financial ratios that are really important from the knowledge perspective. Furthermore, the proposed model computes rather efficiently owing to the high performance computing technology.

Hence, it can be safely concluded that, the developed PTVPSO-FKNN model can serve as a promising alternative early warning system in financial decision-making. Meanwhile, we should note that the proposed model does perform efficiently on the data at hand; however, it is not obvious that the parallel algorithm will lead to significant improvement when applying to the financial data with larger instances. Future investigation will pay much attention to evaluating the proposed model in the larger dataset.

Acknowledgments. This research is supported by the National Natural Science Foundation of China (NSFC) under Grant Nos. 60873149, 60973088, 60773099 and the National High-Tech Research and Development Plan of China under Grant Nos. 2006AA10Z245, 2006AA10A309. This work is also supported

by the Open Projects of Shanghai Key Laboratory of Intelligent Information Processing in Fudan University under the Grand No. IIPL-09-007, the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) and the basic scientific research fund of Chinese Ministry of Education.

References

1. Verikas, A., Kalsyte, Z., Bacauskiene, M., Gelzinis, A.: Hybrid and ensemble-based soft computing techniques in bankruptcy prediction: A survey. Soft Computing-A Fusion of Foundations, Methodologies and Applications 14(9), 995–1010 (2010)
2. Ravi Kumar, P., Ravi, V.: Bankruptcy prediction in banks and firms via statistical and intelligent techniques-a review. European Journal of Operational Research 180(1), 1–28 (2007)
3. Keller, J.: A Fuzzy k-Nearest Neighbor Algorithm. IEEE Transactions on Systems, Man, and Cybernetics 15(4), 580–585 (1985)
4. Bian, H., Mazlack, L.: Fuzzy-rough nearest-neighbor classification approach. In: 22nd International Conference of the North American Fuzzy Information Processing Society, NAFIPS 2003, pp. 500–505. IEEE, Los Alamitos (2003)
5. du Jardin, P.: Predicting bankruptcy using neural networks and other classification methods: The influence of variable selection techniques on model accuracy. Neurocomputing 73(10-12), 2047–2060 (2010)
6. Chapman, B., Jost, G., Van Der Pas, R.: Using OpenMP: portable shared memory parallel programming. The MIT Press, Cambridge (2008)
7. Hart, P.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13(1), 21–27 (1967)
8. Kennedy, J., Eberhart, R., et al.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, vol. 4, pp. 1942–1948 (1995)
9. Ratnaweera, A., Halgamuge, S., Watson, H.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on Evolutionary Computation 8(3), 240–255 (2004)
10. Kennedy, J., Eberhart, R.: A discrete binary version of the particle swarm optimization. In: Proc. Conf. Systems, Man, and Cybernetics, pp. 4104–4108 (1997)
11. Wieslaw, P.: Application of discrete predicting structures in an early warning expert system for financial distress. PhD thesis. Szczecin Technical University, Szczecin (2004)
12. Pietruszkiewicz, W.: Dynamical systems and nonlinear Kalman filtering applied in classification. In: 7th IEEE International Conference on Cybernetic Intelligent Systems, CIS 2008, pp. 1–6. IEEE, Los Alamitos (2008)
13. Salzberg, S.L.: On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery 1(3), 317–328 (1997)
14. Chang, C., Lin, C.: LIBSVM: a library for support vector machines (2001), http://www.csie.ntu.edu.tw/cjlin/libsvm
15. Chang, C., Lin, C., Hsu, C.: A practical guide to support vector classification. Department of Computer Science and Information Engineering, National Taiwan University, Taiwan (2003)
16. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters 27(8), 861–874 (2006)

Semi-supervised Parameter-Free Divisive Hierarchical Clustering of Categorical Data

Tengke Xiong[1], Shengrui Wang[1], André Mayers[1], and Ernest Monga[2]

[1] Department of Computer Science, University of Sherbrooke
[2] Department of Mathematics, University of Sherbrooke
Sherbrooke, QC, Canada, J1K 2R1
{tengke.xiong,shengrui.wang,andre.mayers,ernest.monga}@usherbrooke.ca

Abstract. Semi-supervised clustering can yield considerable improvement over unsupervised clustering. Most existing semi-supervised clustering algorithms are non-hierarchical, derived from the k-means algorithm and designed for analyzing numeric data. Clustering categorical data is a challenging issue due to the lack of inherently meaningful similarity measure, and semi-supervised clustering in the categorical domain remains untouched. In this paper, we propose a novel semi-supervised divisive hierarchical algorithm for categorical data. Our algorithm is parameter-free, fully automatic and effective in taking advantage of instance-level constraint background knowledge to improve the quality of the resultant dendrogram. Experiments on real-life data demonstrate the promising performance of our algorithm.

1 Introduction

Clustering categorical data is more challenging than clustering numeric data due to the lack of an inherently meaningful similarity measure. With large amounts of categorical data being generated in real life, clustering of categorical data has been receiving increasing attention in recent years [3,7,9,10,16,20]. Moreover, in many real-life applications on categorical data, prior knowledge exists in relation to the need or the goal of the data analysis. For example, when a market analyst performs clustering for market segmentation from survey data, he/she will likely want that the customers from the same family be grouped together instead of trivial separation between men and women. None of existing algorithms for clustering categorical data can exploit such prior background knowledge.

Most existing semi-supervised clustering algorithms are non-hierarchical and focus on analyzing numeric data [5]. These non-hierarchical clustering algorithms are derived from the k-means algorithm [2,4,14,18,19]. Besides the k-means variants, labeled instances have been used to help set the parameters of a density-based clustering algorithm in [15], as appropriately setting the parameters is critical but difficult, especially when the density of the clusters differs widely. However, neither the concept of objective function or Euclidean distance used in the non-hierarchical methods nor the density notion in density-based clustering

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 265–276, 2011.

algorithms is naturally meaningful for categorical data. Moreover, most of existing algorithms have some parameters to tune, these parameter-laden algorithms impose our prejudices and presumptions on the data [12].

Incorporating instance-level constraints in hierarchical clustering is more challenging than in partitioning clustering due to the feasibility problem of satisfying the constraints [5,19]. Some studies suggest that hierarchical algorithms can produce better-quality clusters [11,17]. However, little work has been done on the application of background knowledge to hierarchical clustering [4,13], and even these few published hierarchical algorithms are all agglomerative, not divisive. The concept of comparing similarity between pairwise instances, which is used in agglomerative method, is not suitable for categorical data [9,20], thus the error-propagation issue of agglomerative method is even more critical in categorical domain. Furthermore, the high time complexity of agglomerative methods prevents them from being used on very large data sets.

In this paper, we propose several novel concepts and methods suitable for categorical data, based on which, we propose a semi-supervised clustering algorithm for categorical data. We view semi-supervised clustering of categorical data as an optimization problem with extra instance-level constraints. As the optimization problem in the clustering is NP hard [1], thereby we propose a heuristic approach to guide the optimization process to a better solution in terms of satisfying the constraints. We name the new algorithm SDHCC (Semi-supervised Divisive Hierarchical Clustering of Categorical data). SDHCC is systematic and parameter-free. To our knowledge, SDHCC is the first semi-supervised clustering algorithm for categorical data.

2 Instance-Level Constraints

In this paper, prior background knowledge is provided as *must-link* and *cannot-link* constraints on pairs of instances [18,19]. A *must-link* constraint indicates that the two instances have to be in the same cluster, while a *cannot-link* constraint indicates that the two instances must not be placed in the same cluster.

Since *must-link* constraints are equivalence relations, they can be used to generate transitive closures. *Cannot-link* constraints then can be transformed to a *cannot-link* matrix representing link relationship between closures. In this paper, a transitive closure is also called a constraint closure. Besides the transitive closure, a single instance which is not involved in any *must-link* constraint but is involved in a *cannot-link* constraint is also called a constraint closure.

The *cannot-link* matrix is generated from the constraint closures and the *cannot-link* constraints. Let l be the number of constraint closures; then the *cannot-link* matrix M is of order $l \times l$, where $m_{ij} = 1$ if closure c_i and c_j cannot link, otherwise, $m_{ij} = 0$. Closures c_i and c_j cannot link if $\exists X_i \in c_i$ and $\exists X_j \in c_j$ such that (X_i, X_j) is in the set of *cannot-link* constraints. Thus the *cannot-link* matrix M is a symmetric Boolean matrix.

The feasibility of satisfying all constraints for non-hierarchical and agglomerative hierarchical clustering has been studied in [4,5]. The feasibility problem

is more complex in divisive hierarchical clustering, because a divisive method starts with an all-inclusive cluster containing all the instances, and repeatedly chooses one cluster to split into two sub-clusters, which may make violations of *cannot-link* constraints unavoidable lower down (closer to the root) in the clustering hierarchy. In this paper, *must-link* constraints are satisfied at all levels of the clustering tree, whereas *cannot-link* violation is tolerated, especially at the lower levels (closer to the root). Note that we assume the constraints are consistent, dealing with the erroneous constraints is out of the scope of this paper. The following definitions help to measure the degree of *cannot-link* violation in a cluster (under the assumption that all the *must-link* constraints are satisfied).

Definition 1. *Given two constraint closures c_i and c_j, and the cannot-link matrix M, the cannot-link weight of the closure c_i with respect to c_j is*

$$w_i(j) = \begin{cases} |c_j|, & m_{ij} = 1; \\ 0, & m_{ij} = 0. \end{cases}$$

where $|c_j|$ is the number of instances in closure c_j.

Definition 2. *The degree of cannot-link violation of cluster C is defined as:*

$$w(C) = \sum_{c_i \in C} w(c_i|C) = \sum_{c_i \in C} \sum_{c_j \in C} w_i(j) \tag{1}$$

where $w(c_i|C) = \sum_{c_j \in C} w_i(j)$ is the cannot-link weight of c_i in cluster C.

3 The Algorithm

The categorical data set $\chi = \{X_i\}_{i=1}^n$ is represented by an indicator matrix, denoted as Z. Each instance X_i is a multidimensional vector of m categorical attributes with domains $D_1, \cdots, D_m$, respectively. In the new data representation, each categorical value represents a dimension. Let $J = \sum_{t=1}^m |D_t|$ be the total number of categorical values, then the indicator matrix Z is of order $n \times J$, and the general entry of the indicator matrix Z is as follows:

$$z_{ij} = \begin{cases} 1, & \text{if instance } X_i \text{ takes the } j\text{th value } (1 \le j \le J); \\ 0, & \text{otherwise.} \end{cases}$$

In the following presentation of this paper, we use Z_i to denote instance X_i according to the indicator matrix data representation.

We view constraint-free clustering categorical data from an optimization perspective, and propose a novel objective function, i.e., the sum of Chi-square error (SCE). We set the objective of clustering χ into K clusters to minimize SCE, which is defined as follows:

$$SCE = \sum_{k=1}^{K} \sum_{Z_i \in C_k} d_{Chi}(Z_i, C_k) \tag{2}$$

where K is the number of clusters. $d_{Chi}(Z_i, C_k)$ is the Chi-square distance between instance Z_i and Cluster C_k, which is defined as follows:

$$d_{Chi}(Z_i, C_k) = \sum_j \frac{(z_{ij} - \mu_{kj})^2}{\mu_{kj}} \tag{3}$$

Here $\mu_{kj}(1 \leq j \leq J)$ is the jth element of cluster center of cluster C_k, which can be mathematically derived as

$$\mu_{kj} = \left(\frac{1}{|C_k|} \sum_{Z_i \in C_k} z_{ij} \right)^{1/2}. \tag{4}$$

For semi-supervised clustering based on the divisive hierarchical approach, we iteratively deal with the constrained optimization problem with K=2. SDHCC starts with an all-inclusive cluster containing all the categorical instances, and repeatedly chooses one cluster to split into two sub-clusters. Each bisection step consists of three phases: initialization; iterative refinement based on Chi-square distance; and alleviation of the *cannot-link* violation.

3.1 Initialization

The initialization of the bisection consists of two steps, i.e., preliminary splitting based on multiple correspondence analysis (MCA), and redistributing the instances involved in *must-link* constraints to satisfy the *must-link* constraints. The following paragraph describes MCA calculation on indicator matrix of χ (the calculation on a cluster is the same), the interested reader can refer to [8,20] for more details.

Since Z has a total sum of $n \times m$, which is the total number of occurrence of all the categorical values, the correspondence matrix is $P = Z/nm$. Thus the vector of row mass of the correspondence matrix is $r = \frac{1}{n}\mathbf{1}$, the row mass matrix is $D_r = (1/n)I$; the column mass matrix is $D_c = (1/nm)diag(Z^{\mathrm{T}}Z)$, the vector of column mass can be denoted as $c = D_c \times \mathbf{1}$. Under the null hypothesis of independence, the expected value of p_{ij} is $r_i c_j$, and the difference between the observation and the expectation, called the residual value, is $p_{ij} - r_i c_j$. Normalization of the residual value involves dividing the difference by the square root of $r_i c_j$. So the standardized residuals matrix is written as:

$$S = D_r^{-1/2}(P - rc^{\mathrm{T}})D_c^{-1/2} = \sqrt{n}\left(\frac{Z}{nm} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}D_c\right)D_c^{-1/2} \tag{5}$$

Hence, the singular value decomposition (SVD) to compute the residuals matrix (5) is as follows:

$$\sqrt{n}\left(\frac{Z}{nm} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}D_c\right)D_c^{-1/2} = U\Sigma V^{\mathrm{T}} \tag{6}$$

where $U^{\mathrm{T}}U = V^{\mathrm{T}}V = I$. Σ is a diagonal matrix with singular values in descending order: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_s > 0$, where s is the rank of the residuals matrix.

The columns of U are called the left singular vectors, which give us the scale values for the n instances.

The first step of bisection initialization proceeds as follows. To bisect a cluster C_P with $|C_P|$ instances, we apply MCA on the indicator matrix $Z^{(P)}$ of order $|C_P| \times J$ from the $|C_P|$ instances to get the left singular vectors $U^{(P)}$. Each instance Z_i whose first coordinate $U_{i1}^{(P)} \leq 0$ goes to the left child of C_P, which is denoted by C_P^L, and each Z_i whose first coordinate $U_{i1}^{(P)} > 0$ goes to the right child of C_P, which is denoted by C_P^R.

The second step aims to satisfy all the *must-link* constraints. In fact, the preliminary splitting in first step may distribute the instances in a constraint closure in the two sub-clusters. Each closure c split by the first step is re-assigned to the sub-cluster that holds most of its members; i.e., if $|c \cap C_P^L| \geq |c \cap C_P^R|$, the closure c goes to the left child C_P^L, otherwise, the closure c goes to the right child C_P^R.

3.2 Refinement

In the refinement phase, we employ the Chi-square distance to measure the dissimilarity between a single instance, as well as a constraint closure, and a cluster. In fact, after re-assignment of the second step of the initialization phase, the instances from each closure are assembled together, so the instances in a closure will be treated as an entity in performing the refinement. Therefore, we define the dissimilarity between a closure and a cluster of categorical instances as follows.

Definition 3. *The dissimilarity between a constraint closure c and a cluster C_i is the Chi-square distance between them, i.e.,*

$$d_{Chi}(c, C_i) = \sum_j \frac{(\bar{c}_j - \mu_{ij})^2}{\mu_{ij}} \tag{7}$$

where c is in cluster C_i, and $\bar{c}_j (1 \leq j \leq J)$ is the jth element of the center of closure c, i.e., $\bar{c}_j = \left(\frac{1}{|c|} \sum_{Z_i \in c} z_{ij}\right)^{1/2}$. When c is a singleton closure, Formula (7) is the same as (3).

Theorem 1. *In a refinement phase (not limited to bisection operation of hierarchical clustering), deciding the membership of a closure according to the Chi-square distance defined in (7) is equivalent to making the decision according to the sum of the Chi-square distance of all the instances in the closure, which is defined in (3).*

The proof of Theorem 1 is omitted here due to lack of space.

Definition (3) and Theorem (1) provide a significant computational advantage. They show that it is only necessary to store the statistic features of each constraint closure and those of the two resulting sub-clusters, to speed up the refinement process. In other words, the instances in a closure are treated as an entity, the number of instances for refinement is thus reduced. The cluster features of C_P have two elements: one is the J-dimensional vector of numbers of

occurrences of all the categorical values, denoted as NO_P; and the other is the number of instances in the cluster, denoted as $|C_P|$. The features of a closure are the same as those of a cluster. We do not need to revisit all the instances in the cluster to update the cluster features; instead, we just need to record the instances which were relocated to do the update, so the cluster features can thus be updated efficiently after each iteration cycle.

The statistic feature of Chi-square distance requires that individual instance (or closure) be in the measured set. When calculating the distance between an instance Z_i and cluster C_P, where Z_i is not in C_P, we should proceed as if Z_i is in cluster C_P by updating the cluster features to $r_P \times NO_P + Z_i$ and $r_P \times |C_P| + 1$, and after that, restore the cluster features. Here r_P is the balance ratio of cluster size of the two resulting sub-clusters. The balance ratio is set to 1.0 for C_P^L, and $|C_P^L|/|C_P^R|$ for C_P^R if $|C_P^L| \geq |C_P^R|$, vice versa if $|C_P^L| \leq |C_P^R|$. The same applies to the distance calculation between a closure c and C_P, where c is not in C_P. We use the balance ratio to solve the bias issue in the relocation process. This is because that the rare categorical value of the individual instance/closure has relative high frequency in smaller cluster, which leads to smaller Chi-square distance.

The refinement of the instances in C_P^L and C_P^R proceeds as in Algorithm 1.

Algorithm 1. Refinement after initialization

1. Calculate the cluster features of C_P^L and C_P^R.
2. For each instance Z_i in C_P^L
   if $Z_i \in c$ {if $d_{Chi}(c, C_P^R) < d_{Chi}(c, C_P^L)$, move c to C_P^R;}
   else if $d_{Chi}(Z_i, C_P^R) < d_{Chi}(Z_i, C_P^L)$, move Z_i to C_P^R;
   For each instance Z_i in C_P^R
   if $Z_i \in c$ {if $d_{Chi}(c, C_P^L) < d_{Chi}(c, C_P^R)$, move c to C_P^L;}
   else if $d_{Chi}(Z_i, C_P^L) < d_{Chi}(Z_i, C_P^R)$, move Z_i to C_P^L;
3. Update the cluster features of NO_P^L, $|C_P^L|$ and NO_P^R, $|C_P^R|$.
4. Repeat steps (2) and (3) until the membership no longer changes.

3.3 Alleviation of *Cannot-Link* Violation

In this subsection, a novel divide-and-merge method is proposed to alleviate the *cannot-link* violation in each bisection. It proceeds as follows (The pseudo-code is given in Algorithm 2):

Dividing operation: in each sub-cluster (C_P^L or C_P^R), if it has a *cannot-link* constraint, divide it into two sets. One of these, called the *alien* set, contains the target constraint closure which is most dissimilar with the sub-cluster, and also the instances that are similar to the target closure (how to choose the target closure will be presented afterwards); the other set, which is called the *native* set, contains the rest of the instances in the sub-cluster. The pseudo-code for the dividing operation is shown in Algorithm 3.

Merging operation: After dividing operation, the alien set is merged with the native set of the other sub-cluster if doing so can decrease the sum of the degree of *cannot-link* violation of C_P^L and C_P^R. The pseudo-code for the merging operation is shown in Algorithm 4.

Algorithm 2. Alleviating the *cannot-link* violation

1. if C_P^L has *cannot-link* violation
   find target closure c_t^L;
   Divide-Cluster(C_P^L,c_t^L); //divide C_P^L into C_P^{LL} and C_P^{LR}, $c_t^L \subseteq C_P^{LR}$
   else $C_P^{LL} = C_P^L$; $C_P^{LR} = \emptyset$;
2. if C_P^R has *cannot-link* violation
   find target closure c_t^R;
   Divide-Cluster(C_P^R,c_t^R); //divide C_P^R into C_P^{RL} and C_P^{RR}, $c_t^R \subseteq C_P^{RR}$
   else $C_P^{RL} = C_P^R$; $C_P^{RR} = \emptyset$;
3. if($C_P^{LR} \neq \emptyset$ or $C_P^{RR} \neq \emptyset$)
   Merge-Clusters(C_P^{LL},C_P^{LR},C_P^{RL},C_P^{RR});
4. Repeat steps (1), (2) and (3) until no target closure can be found to decrease the sum of the degree of *cannot-link* violation of C_P^L and C_P^R.
5. Recall Algorithm 1.

Algorithm 3. Divide-Cluster
Input: Cluster C, target closure c_t, $c_t \subset C$
Output: Cluster C^L, C^R, $C^L \cup C^R = C$; $C^L \cap C^R = \emptyset$; $c_t \subseteq C^R$

1. Initialize cluster $C^t = C$
2. Repeat until termination condition is satisfied
   (a) Initialize bisection of C^t;
   (b) recall Algorithm 1 to refine the instances in C^{tL} and C^{tR};
   (c) if ($c_t \subseteq C^{tL}$), $C^t = C^{tL}$; else $C^t = C^{tR}$;
3. $C^R = C^t$; $C^L = C - C^R$

Algorithm 4. Merge-Clusters
Input: Clusters C_P^{LL},C_P^{LR},C_P^{RL},C_P^{RR}
Output: Cluster C_P^L, C_P^R
if $w(C_P^{LL} \cup C_P^{RR}) + w(C_P^{RL} \cup C_P^{LR}) < w(C_P^L) + w(C_P^R)$
  $C_P^L = C_P^{LL} \cup C_P^{RR}$; $C_P^R = C_P^{RL} \cup C_P^{LR}$.
else if $w(C_P^L \cup C_P^{RR}) + w(C_P^{RL}) < w(C_P^L) + w(C_P^R)$
  $C_P^L = C_P^L \cup C_P^{RR}$; $C_P^R = C_P^{RL}$.
else if $w(C_P^R \cup C_P^{LR}) + w(C_P^{LL}) < w(C_P^L) + w(C_P^R)$
  $C_P^L = C_P^{LL}$; $C_P^R = C_P^R \cup C_P^{LR}$.

The closure which has the largest *cannot-link* weight in the sub-cluster is chosen as the target closure; if in a sub-cluster there is more than one closure with the largest *cannot-link* weight, choose the one with the greatest Chi-square distance from the cluster. To use the constraint knowledge to guide the assignment

of unconstrained instances, the unconstrained instances which are similar with the target closure are removed together. The divide-and-merge operation on C_P^L and C_P^R proceeds iteratively until the sum of the degree of *cannot-link* violation of C_P^L and C_P^R cannot decrease.

We make use of the global clustering quality measure proposed in [3] to decide when to terminate splitting in Algorithm 3 and the splitting process in constructing the clustering tree. The termination condition is that either of the two conditions given below is satisfied:

1. Algorithm 1 ends with one cluster, which means that the relocation algorithm finally converges into placing all the instances into one cluster.
2. Clustering quality does not increase, i.e., $Q(\{C^t\}) \geq Q(\{C^{tL}, C^{tR}\})$, and neither C^{tL} nor C^{tR} violates the *cannot-link* constraints.

In the condition (2), $Q(\Gamma)$ denotes the global clustering quality measure [3].

4 Experimental Results

In this section, we study the performance of SDHCC on real data sets from the UCI Machine Learning Repository and 20-Newsgroup text data. We implemented a semi-supervised clustering algorithm for categorical data by combining the semi-supervised algorithm COP-KMEANS in [19] and the k-modes algorithm for categorical data in [16]; the combination is called SKModes in this paper. We also compared our algorithm with SKModes, as well as the state-of-the-art semi-supervised clustering algorithms for numeric data, which are constrained agglomerative hierarchical clustering algorithm in [5], named AggHie in this paper, and the algorithm based on Hidden Markov Random Fields [2], named HMRF-Kmeans in this paper. Both complete version (named SDHCC-M-C) and ablated version (named SDHCC-M) of SDHCC are used for comparison. In SDHCC-M, the *cannot-link* constraints are only used in the termination condition to justify whether a cluster should be split or not. We investigate the effectiveness of using instance-level constraint knowledge by comparing the clustering results of the semi-supervised algorithms with those of its underlying unsupervised algorithms.

In our performance evaluation, we adopt the F-measure as the clustering validation measure. In hierarchical clustering, the maximum is taken over all clusters i at all levels; in partitioning clustering, the maximum is taken over the K clusters. To evaluate the performance of the algorithms objectively, we used twofold cross-validation on each data set for 20 trials. In each trial, we randomly selected 50% of the instances as the test set, and the F-measure was calculated only on the test set. The remaining half of the data set was used as a training set, and the constraints were generated by randomly selecting pairs of instances from the training set, creating a *must-link* constraint if the pair instances have the same label and a *cannot-link* constraint if they have different labels. The clustering algorithms were run on the whole data set, and the results given are the averages of the results of the 20 trials.

4.1 Data Sets

The four UCI data sets used in this paper are the *Zoo*, Congressional Voting Records (*Votes*), Wisconsin breast cancer (*Cancers*) and *Mushroom* sets. The numbers of categorical values for these sets are J=36,48,91 and 117 respectively.

We also extracted two data sets from the 20-Newsgroup data, and both data sets are preprocessed by removing stop-words and very high-frequency and low-frequency words, the same as the methodology used in [6]. After preprocessing, each data set is prepared with two versions, one is numeric data using TF-IDF weighting method, for details, refer to [6]; the other is categorical (transactional) data, where each article is simply represented by collection of the words appearing in the article (after preprocessing). The description of the two data sets is as follows.

Similar-2: this data set consists of 200 articles from two similar topics, i.e., comp.sys.ibm.pc.hardware and comp.sys.mac.hardware, and each topic has 100 articles. The data is represented in 1233 dimensions (words), and in the categorical version, $J = 1233 \times 2$ as each word is analogous to one categorical attribute which takes two values indicating inclusion or non-inclusion of the word.

Different-3: this data set consists of 300 articles from three different topics, i.e., alt.atheism, comp.sys.ibm.pc.hardware and talk.politics.mideast, and each topic has 100 articles. The data is represented in 2470 dimensions (words), thus $J = 2470 \times 2$ in the categorical version.

4.2 Results and Discussion

Figure 1, 2, 3 and 4 show the clustering results on the *Zoo*, *Votes*, *Cancers*, and *Mushroom* data sets respectively. For SKModes, we set K=7 on Z*oo*; K=3 on *Votes* and *Cancers*, as it fails to generate a clustering when the number of constraints is greater than 100 if we set K=2; K=4 on *Mushroom* for the same reason. For HMRF-Kmeans, we set K to the number of classes on the four data sets. AggHie and HMRF-Kmeans run on the Boolean indicator matrix of the categorical data sets.

From these figures we can see that the use of instance-level prior knowledge produces a remarkable improvement in SDHCC, except on the *Cancers* data set. The unsupervised algorithm (DHCC) already yields outstanding clustering results on *Cancers*, with an F-measure attaining 0.97; thus, the instance-level constraints do not provide much informative knowledge to guide the clustering process. On the *Zoo* and *Votes* data sets, SDHCC-M-C and SDHCC-M show quite comparable clustering performance, while on the *Mushroom* data set, which is of higher dimension and much larger size, SDHCC-M-C outperforms SDHCC-M by a wide margin. For other three algorithms, none of them can reap potential benefit of prior knowledge on the four UCI data sets, except SKModes on the simple *Zoo* data set. Especially for AggHie, the clustering performance deteriorates when the instance-level constraints are incorporated.

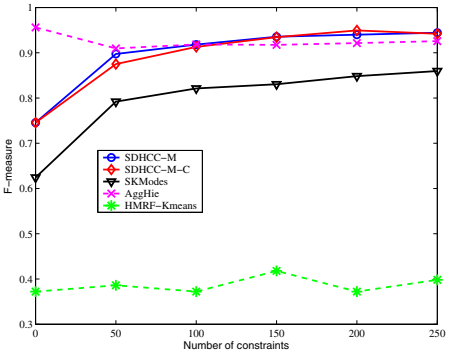


Fig. 1. Clustering results on *Zoo*

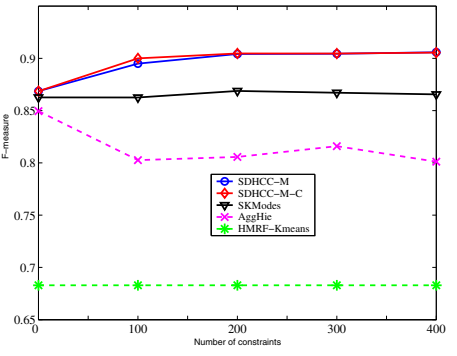


Fig. 2. Clustering results on *Votes*

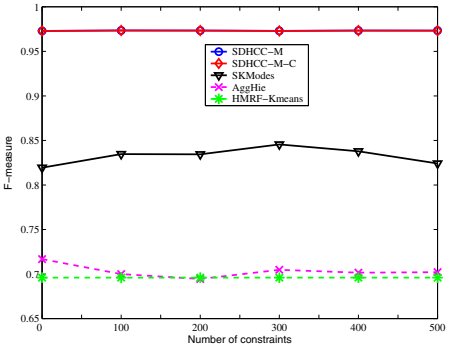


Fig. 3. Clustering results on *Cancers*

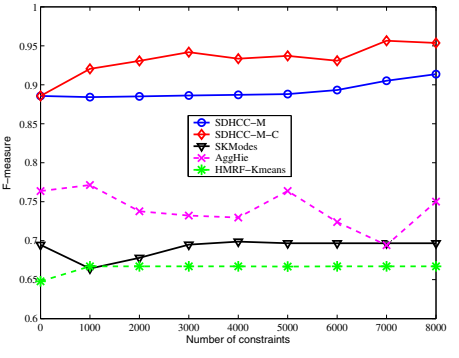


Fig. 4. Clustering results on *Mushroom*

Overall, our complete SDHCC algorithm shows superior clustering performance on the UCI real-life data. The values of F-measure on the four data sets are all greater than 0.9, which means the clusters discovered by SDHCC-M-C closely match the natural classification.

Figure 5 and 6 show the clustering results on *Similar-2* and *Different-3* data sets respectively. We set K=2 on *Similar-2* and K=3 on *Different-3* for SKModes and HMRF-Kmeans. In these figures, AggHie-N and HMRF-Kmeans-N indicate the results on numeric data, while AggHie-C and HMRF-Kmeans-C indicate the results on categorical data.

From these figures we can see that our algorithm can reap remarkable improvement even on the succinct categorical representation of text data, especially on the *Different-3* data set. *Similar-2* data impose more challenge than *Different-3* on clustering algorithms, since the overlap between the two topics in *Simila-2* is significant. SDHCC-M-C outperforms its ablated version SDHCC-M on the text data.

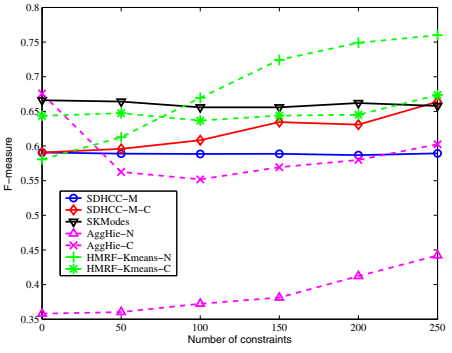


Fig. 5. Clustering results on *Similar-2*

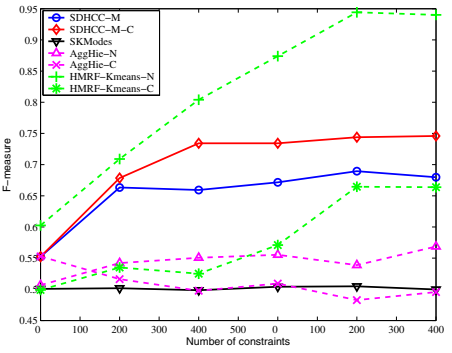


Fig. 6. Clustering results on *Different-3*

HMRF-Kmeans demonstrates superior performance on numeric TF-IDF data; however, it gains much less improvement on the categorical data. SDHCC-M-C outperforms HMRF-Kmeans by a wide margin on categorical version of *Different-3* data set. On the categorical *Similar-2* data, SDHCC-M-C performs best in terms of the improvement of clustering quality. AggHie reaps potential benefit from prior knowledge on numeric TF-IDF data, however, its performance on categorical version deteriorates when the instance-level constraints are incorporated, the same as it does on the UCI data set. For SKModes, the prior instance-level knowledge has little influence on the clustering performance.

5 Conclusions

In this paper, we have proposed a semi-supervised divisive hierarchical clustering algorithm for categorical data (SDHCC).We formalize clustering categorical data as optimizing the objective function, and exploit pairwise *must-link* and *cannot-link* constraint knowledge to guide the optimization process to a better solution in terms of satisfying the constraints, which would also be beneficial to the unconstrained instances. Experimental results on UCI data sets and 20-Newsgroup text data demonstrate that our semi-supervised algorithm shows remarkable improvement over the unsupervised clustering algorithm DHCC. Most important, our semi-supervised clustering algorithm could take advantage of the potential improvement from a small amount of knowledge, which is very useful in real applications, as it is very expensive to provide large numbers of pairwise constraints from human experts.

Our experiments also show that the mainstream semi-supervised clustering algorithms for numeric data, such as AggHie and HMRF-Kmeans, are not suitable for categorical data. Incorporating the instance-level constraint knowledge even could deteriorate the clustering quality of the underlying unsupervised clustering algorithm.

References

1. Aloise, D., Deshpande, A., Hansen, P., Popat, P.: NP-hardness of Euclidean sum-of-squares clustering. Mach. Learn. 75, 245–248 (2009)
2. Basu, S., Bilenko, M., Mooney, R.J.: A Probabilistic Framework for Semi-Supervised Clustering. In: ACM KDD (2004)
3. Cesario, E., Manco, G., Ortale, R.: Top-down parameter-free clustering of high-dimensional categorical data. IEEE T. Knowl. Data En. 19 (2007)
4. Davidson, I., Ravi, S.S.: Clustering with constraints: Feasibility issues and the k-means algorithm. In: SIAM SDM (2005)
5. Davidson, I., Ravi, S.S.: Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results. Data Min. Knowl. Disc. 18 (2009)
6. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Mach. Learn. 42, 143–175 (2001)
7. Gan, G., Wu, J.: Subspace clustering for high dimensional categorical data. SIGKDD Explorations 6 (2004)
8. Greenacre, M., Blasius, J.: Multiple Correspondence Analysis and Related Methods. Chapman & Hall, Boca Raton (2006)
9. Guha. S., Rastogi, R., Shim, K.: ROCK: a robust clustering algorithm for categorical attributes. In: IEEE ICDE (1999)
10. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical value. Data. Min. Knowl. Disc. 2, 283–304 (1998)
11. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Comput Surv. 31 (1999)
12. Keogh, E., Lonardi, S., Ratanamahatana, C.: Toward parameter-free data mining. In: ACM KDD (2004)
13. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering. In: ICML, pp. 307–314 (2002)
14. Kulis, B., Basu, S., Dhillon, I., Mooney, R.: Semi-Supervised graph clustering: a Kernel Approach. Mach. Learn. 74, 1–22 (2009)
15. Lelis, L., Sander, J.: Semi-Supervised Density-Based Clustering. In: ICDM (2009)
16. San, O.M., Huynh, V., Nakamori, Y.: An alternative extension of the k-means algorithm for clustering categorical data. Int. J. Appl. Math. Comp. 14 (2004)
17. Tan, P., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2005)
18. Tang, W., Xiong, H., Zhong, S., Wu, J.: Enhancing Semi-Supervised Clustering: A Feature Projection Perspective. In: ACM KDD (2007)
19. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: ICML, pp. 577–584 (2001)
20. Xiong, T., Wang, S., Mayers, A., Monga, E.: A New MCA-based Divisive Hierarchical Algorithm for Clustering Categorical Data. In: IEEE ICDM (2009)

Identifying Hidden Contexts in Classification

Indrė Žliobaitė

Eindhoven University of Technology, Eindhoven, The Netherlands
Smart Technology Research Center, Bournemouth University, Poole, UK
zliobaite@gmail.com

Abstract. In this study we investigate how to identify hidden contexts from the data in classification tasks. Contexts are artifacts in the data, which do not predict the class label directly. For instance, in speech recognition task speakers might have different accents, which do not directly discriminate between the spoken words. Identifying hidden contexts is considered as data preprocessing task, which can help to build more accurate classifiers, tailored for particular contexts and give an insight into the data structure. We present three techniques to identify hidden contexts, which hide class label information from the input data and partition it using clustering techniques. We form a collection of performance measures to ensure that the resulting contexts are valid. We evaluate the performance of the proposed techniques on thirty real datasets. We present a case study illustrating how the identified contexts can be used to build specialized more accurate classifiers.

1 Introduction

In classification tasks some variables directly predict the class label, others can describe context. Contexts are artifacts in the data, which do not directly predict the class label, like accent in speech recognition. Taking contexts into the learning process can help to build more specialized and accurate classifiers [2], solve sample selection bias [12], concept drift [17] problems.

Context may not necessarily be present in a form of a single variable in the feature space. To recover *hidden contexts* the input data can be clustered [5, 9, 15]. The problem is, that clustering can capture some class label information, which would shade away the contexts. Consider a diagnostics task, where patient tests are taken by two pieces of equipment, which are calibrated differently. If we cluster patient data, the resulting clusters might correspond to 'healthy' and 'sick' (which are the classes) or 'sample taken by equipment A' and 'sample taken by equipment B' (which is a context), but likely a mix of both. Thus, to capture context specific information, we intend to force independence between contexts and class labels. In addition, capturing noise is undesired, therefore the resulting contexts need to be non-random and stable.

Context identification has predictive and descriptive goals. Grouping the data provides an opportunity to achieve more accurate classification employing context handling strategies, as well as better understand the phenomenon behind

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 277–288, 2011.

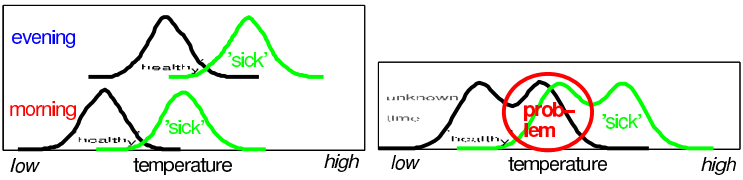


Fig. 1. An example of two contexts

the data. Context identification can be considered as a preprocessing step in supervised learning, like feature selection, instance selection, or recovering missing values. We aim for a filter approach, where contexts are generic, not tied with a particular handling strategy.

In this study we propose three techniques for identifying hidden contexts, which force independence between the contexts and class labels. The objective is to output an explicit grouping of the data. We require the grouping to ignore the class label information. Thus, we aim to hide class discriminatory information before partitioning. These techniques can be used in different context handling strategies or for forming new ones.

We analyze the performance of the proposed techniques on thirty real datasets. We also present a case study, which illustrates one example strategy for handling the identified contexts in classification.

The paper is organized as follows. Section 2 defines context. Section 3 discusses related work. In Section 4 we propose three techniques for identifying hidden contexts. Sections 5 and 6 present experimental evaluation. Section 7 concludes the study.

2 Problem Set-Up

Consider a classification problem in p-dimensional space. Given a set of instances with labels ($\mathbf{X} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$) the task is to produce a classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$. In this study we define context as a secondary label z of an instance X, which is independent from the class label y, but can explain the class label better when used together with the predictive features. That is, $p(y|z) = p(y)$, but $p(y|X^*, z) \neq p(y|X^*)$, where $X^* \subseteq X$. Context might be expressed as a variable in the feature space (known context) or as a latent variable (unknown context).

Consider as a toy example a task, where a patient is diagnosed 'sick' or 'healthy' based on the body temperature. It is known that in the evening people tend to have higher temperature *independently* of being sick or healthy. If we know the context, i.e. whether the temperature was measured in the morning or in the evening, diagnostic task is easy, as illustrated in Figure 1. However, if the time is unknown, then diagnosing becomes problematic. The time itself is independent from the class label, stand alone it does not diagnose.

Some more examples of context include accent in speech recognition, light in image recognition, seasonality in sales prediction, weekday in electricity load or bus travel time prediction, industry crisis in credit scoring.

The context label may be *directly observable* or *hidden*, depending on the application. Hidden context variable z is not explicitly present as a feature x in the feature space $\mathcal{X}$, but information about it is assumed to be captured in the feature space $\mathcal{X}$, i.e. $z = f(X)$, $X \in \mathcal{X}$. An example of directly observable context is time. Customer segments in marketing tasks or bankruptcy prediction represent hidden context. Different segments might have different behavior.

Evaluation of the identified contexts is not straightforward. Different context handling strategies can lead to different gains or losses in the classification accuracies, which are not necessarily due to good or bad context identification. We require the resulting contexts to be independent from the class labels, valid (not random grouping of the data) and stable on random subsamples of the same data. The criteria to measure these aspects are formulated in Section 5.

3 Positioning within Related Work

Context-awareness is widely used in ubiquitous and pervasive computing to characterize the environmental variables [14]. In machine learning the term usually characterizes the features that do not determine or influence the class of an object directly [2, 15]. A wide body of literature on concept drift considers only time context [7, 17]. Typically contexts assumed to be known. Mixture models (see [4]) can be considered as an approach to identify hidden contexts.

Our context identification techniques are novel as they force independency from the class labels. A need for such approaches was mentioned before in a light of context handling strategies [16] and multiple classifier systems [3]. Turney [16] formulated the problem of recovering implicit context information and proposed two techniques: input data clustering and time sequence (which we leave out of the scope assuming that the chronological order is unknown). Turney expressed a concern that clustering might capture class label information and indicated a need for further research, our work can be seen as a follow up.

A recent work by Dara et al [3] explores the relation between the characteristics of data partitions and final model accuracy in multiple classifier systems. The work experimentally confirms the benefits of partitions which are not correlated with the class labels. These results support the motivation of our work.

As a result of their analysis Dara et al [3] propose a semi-randomized partitioning strategy to cluster and then swap some instances across the clusters, which can be seen as a mixture of clustering and boosting. We excluded this strategy from our investigations after preliminary experiments, since even though it pushes towards independence in class labels within the partitions (which is our objective as well), due to randomization the procedure of assigning an unseen instance to one of the partitions can no longer be deterministic.

Extracting hidden context is related to the context handling strategies [16]. The strategies are not limited to building a separate classifier or a combination for each context. Contextual information can be also used to adjust the input data, model parameters or model outputs. Analysis of the performance of different handling strategies is out of the cope of this study. The identified and validated contexts can be used as building blocks to handling strategies.

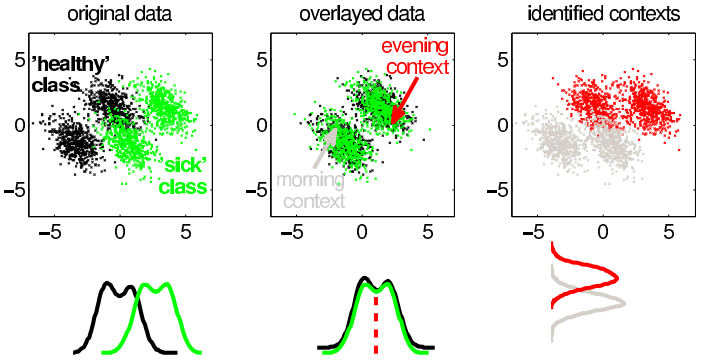


Fig. 2. An illustration of *Overlay* technique

4 Three Techniques for Identifying Hidden Contexts

In this section we present techniques for identifying hidden contexts. Given a dataset, the task is to allocate the instances into k groups, so that the data within the groups is related, but the groups are not related to the class labels.

Context identification techniques require two mechanisms: (1) how to group the training data $\mathbf{X}$ into k contexts and (2) how to assign an unseen instance $X \notin \mathbf{X}$ to one of the contexts.

Clustering (CLU) of the input data is the baseline technique to identify hidden contexts, when building local models [5, 9–11]. The procedure is summarized in Figure 4. Clustering captures closeness of the data instances in the feature space. For classification tasks the feature space is typically formed with an intention to predict the class labels. If class membership information is strongly present in the data, clustering is likely to capture it as well.

To overcome this issue we propose **Overlay (OVE)** technique. To hide the label information we move the classes on top of each other, as illustrated in Figure 2, by normalizing each class to zero mean. The technique assumes that class discriminatory information lies in the class means. We cluster the overlayed data to extract contexts. Unfortunately, for the incoming new data we cannot do overlay, because the labels are unknown. We solve this by introducing a supervised context learning. Given the instances $\mathbf{X}$ we treat the obtained contexts $\mathbf{z}$ as labels and learn a classifier $z = \mathcal{K}_{OVE}(X)$. We use the diagonal linear discriminant [4] as a classifier $\mathcal{K}_{OVE}$. The procedure is summarized in Figure 4.

Overlay technique is based on the assumption that the class distributions are symmetric across different contexts, which often might not be true. We generalize Overlay by introducing **Projection (PRO)** technique, which rotates the data to hide the class label information. The idea is opposite to Linear Discriminant Analysis (LDA) [4]. The goal is to find a transformation that minimizes between-class variance and maximizes within-class variance, see Figure 3.

We seek to find a transformation $\mathbf{w}$ to obtain a projection $\check{\mathbf{x}} = \mathbf{w}'\mathbf{X}$. Within-class c_i covariance is $s_i^2 = \sum_{y_j=c_i}(\mathbf{X}_j - \mu_i)(\mathbf{X}_j - \mu_i)'$, where μ_i is the class

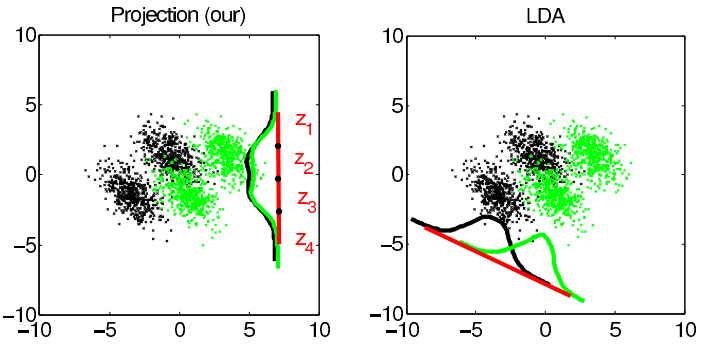


Fig. 3. An illustration of *Projection* technique

mean. The total within class covariance is $S_s := s_1^2 + s_2^2 + \ldots + s_c^2$. Between-class covariance is $S_b := \frac{1}{c}\sum_{i=1}^{c}(\mu_i - \mu)(\mu_i - \mu)'$, where μ is the mean of all the data.

In LDA Fisher criterion $J(w) = \frac{\mathbf{w}' S_b \mathbf{w}}{\mathbf{w}' S_s \mathbf{w}}$ is maximized. We minimize it. The problem transforms into eigenvalue decomposition $S_s^{-1} S_b \mathbf{w} = \lambda \mathbf{w}$. We choose the eigenvector $\mathbf{w}$ corresponding to the smallest eigenvalue $\min \lambda$. To determine contexts $\mathbf{z}$, we transform the training data into 1D space $\breve{\mathbf{x}} = \mathbf{w}'\mathbf{X}$ and simply split the range of values into k equal intervals (like slicing a loaf of bread). An unseen instance X is transformed into 1D $\breve{x} = \mathbf{w}X$ and assigned a context, based on the interval, to which it falls into. The procedure is presented in Figure 4.

In addition to Overlay and Projection, we explore **Feature underselection (FUS)** technique, which discards the features, that are the most correlated with the class label, and clusters the remaining features. It is described in Figure 4.

All the presented techniques assume that k is given. In the case study (Section 6) we will show, how k can be determined using the stability criterion.

5 Experimental Evaluation

The goal of the experiments is to compare the introduced techniques in terms of the desired properties: not to capture the class labels, at the same time controlling, that the resulting partitions are valid and stable.

5.1 Evaluation Criteria

To measure the three desired characteristics (independence from the class labels, validity and stability) we adopt metrics commonly used in clustering evaluation.

For **measuring independence** between the labels and the identified contexts, we employ *Normalized Mutual Information* (NMI), which is widely used to assess clustering performance [18]. It evaluates the purity of clusters with respect to class labels. For the context identification task *low* NMI is desired. For two random variables $\mathbf{y}$ and $\mathbf{z}$ $NMI(\mathbf{y}, \mathbf{z}) = I(\mathbf{y}, \mathbf{z}) / \sqrt{H(\mathbf{y})H(\mathbf{z})}$, where $I(\mathbf{y}, \mathbf{z})$ is the mutual information, $H(\mathbf{x})$ and $H(\mathbf{z})$ are the respective entropies. Note,

CLUSTERING (CLU)
input: dataset $\mathbf{X}$; number of contexts k.
output: context labels $\mathbf{z}$; the rule for assigning unseen instances$z = \mathcal{K}_{CLU}(X)$.

1. Cluster the dataset to obtain contexts $\mathbf{z} = clust(\mathbf{X}, k)$, where $clust(., k)$ is any distance based clustering algorithm.
2. Fix the rule for unseen instances $\mathcal{K}_{CLU}(X) : z = \arg\min_{i=1..k} dist(X, C_i)$, where $C_1, \ldots, C_k \in \mathcal{X}$ are the resulting cluster centers and $dist()$ is a distance function corresponding to the chosen clustering algorithm.

OVERLAY (OVE)
input: labeled dataset $(\mathbf{X}, \mathbf{y})$; number of contexts k.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{OVE}(X)$.

1. Split $\mathbf{X}$ into c groups: $X_j \in \mathbf{X}_i$ if $y_j = i, \forall X \in \mathbf{X}$, c is the number of classes.
2. Shift each class to *zero* mean: for $i = 1 \ldots c$ $\hat{\mathbf{X}}_i = \mathbf{X}_i - \mu_i$, where μ_i is the mean of class c_i.
3. Overlay the classes $\hat{\mathbf{X}} = \{\hat{\mathbf{X}}_1 \cup \ldots \cup \hat{\mathbf{X}}_c\}$.
4. Cluster $\hat{\mathbf{X}}$ to obtain the contexts $\mathbf{z} = clust(\hat{\mathbf{X}}, k)$.
5. Learn a classifier $z = \mathcal{K}_{OVE}(X)$ using $\mathbf{X}$ as input data and $\mathbf{z}$ as labels.

PROJECTION (PRO)
input: labeled dataset $(\mathbf{X}, \mathbf{y})$; number of contexts k.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{PRO}(X)$.

1. Find a transformation vector $\mathbf{w}$, corresponding to the smallest eigenvalue $\min \lambda$ in $S_s^{-1} S_b \mathbf{w} = \lambda \mathbf{w}$, where S_s is within-class covariance and S_b is between-class covariance.
2. Transform into 1D space: $\breve{\mathbf{x}} = \mathbf{w}'\mathbf{X}$.
3. For $j = 1 \ldots k$ find the intervals $r_j = \breve{x}_{min} + q(j-1)$, where $q = (\breve{x}_{max} - \breve{x}_{min})/k$.
4. Find context labels $z = j | \breve{x} \in r_j, \forall \breve{x} \in \breve{\mathbf{x}}$.
5. Fix the rule $\mathcal{K}_{PRO}(X) : z = j | \breve{x} \in r_j$, $\breve{x} = w'X$.

FEATURE UNDERSELECTION (FUS)
input:labeled dataset $(\mathbf{X}, \mathbf{y})$; a number of contexts k; number of features to select m.
output: context $\mathbf{z}$; rule for unseen instances $z = \mathcal{K}_{FUS}(X)$.

1. For $i = 1 \ldots p$ find $\rho_i = corr(\mathbf{x}_i, \mathbf{y})$, where p is the dimensionality, $\mathbf{x}_i$ is the i^{th} dimension of the data.
2. Sort correlations: $|\rho_{i1}| \leq |\rho_{i2}| \leq \ldots \leq |\rho_{ip}|$.
3. Pick m dimensions, the least correlated with class labels: $\tilde{\mathbf{X}} = (\mathbf{x}_{i1}\mathbf{x}_{i2} \ldots \mathbf{x}_{im})'$.
4. Cluster $\tilde{\mathbf{X}}$ to obtain the contexts $\mathbf{z} = clust(\tilde{\mathbf{X}}, k)$.
5. Fix the rule $\mathcal{K}_{FUS}(X) : z = \arg\min_{i=1..k} dist(\tilde{X}, \tilde{C}_i)$, where $\tilde{X} = (x_{i1}x_{i2} \ldots x_{im})'$ and $\tilde{C}_1, \ldots, \tilde{C}_k \in \tilde{\mathcal{X}}$.

Fig. 4. Techniques for finding hidden contexts

that $NMI \in [0,1]$ and $NMI(\mathbf{y},\mathbf{y}) = 1$. Given the context assignments $\mathbf{z}$ and the respective class labels $\mathbf{y}$, the NMI is estimated [18] as

$$NMI(\mathbf{y},\mathbf{z}) = \frac{\Sigma_{l=1}^{k}\Sigma_{h=1}^{c} N_{lh} \log \frac{nN_{lh}}{n_l \hat{n}_h}}{\sqrt{(\Sigma_{l=1}^{k} n_l \log \frac{n_l}{n})(\Sigma_{h=1}^{c} \hat{n}_h \log \frac{\hat{n}_h}{n})}}, \quad (1)$$

where n_l is the number of data instances contained in the cluster C_l $(1 \leq l \leq k)$, $\hat{n}_h$ is the number of instances belonging to the class h $(1 \leq h \leq c)$, N_{lh} is the number of instances that are in the intersection between the cluster C_l and the class h, and n is the total number of instances.

Measuring validity. If we optimized only NMI, assigning instances to contexts at random would be the optimal solution. To control that the identified contexts are not random, we require the identified context labels to be learnable from the data. We use the Naive Bayes (NB) classifier. $VAL(\mathbf{z}|\mathbf{X})$ is the error rate of NB using 10 fold cross validation, *the smaller* the better. We normalize it w.r.t. random assignment of contexts $NVAL(\mathbf{z}|\mathbf{X}) = VAL(\mathbf{z}|\mathbf{X}) / VAL(\phi(k)|\mathbf{X})$, where $\phi(k)$ is a set of contexts (k) assigned at random, thus $NVAL \in [0,1]$.

Measuring stability. In addition to independence and validity we want to minimize the chance of overfitting the training data, which we measure using the stability index for clustering proposed in [13]. The dataset $\mathbf{X}$ is at random split into two sets of equal size $\{\mathbf{X_u} \cup \mathbf{X_v}\} = \mathbf{X}$. Each subset is clustered using the same clustering algorithm $\mathbf{u} = clust_u(\mathbf{X}_u)$, $\mathbf{v} = clust_v(\mathbf{X_v})$, $clust_u()$ and $clust_v()$ denotes fixed parameterizations resulting after clustering (e.g. cluster centers in k-means). Then the fixed $clust_v()$ is applied to the subset $\mathbf{X_u}$ to obtain alternative cluster assignment $\grave{\mathbf{u}} = clust_v(\mathbf{X_u})$. If clustering is stable, given a correct permutation $u^* = map(u)$ of cluster labels $\grave{\mathbf{u}}$ and $\mathbf{u}^*$ should be the same. The (in)stability index is the share of different cluster assignments $STA(\mathbf{u},\grave{\mathbf{u}}) = \frac{1}{n}\Sigma_{i=1}^{n}\mathbf{1}(u_i = map(\grave{u}_i))$, where $\mathbf{1}() = \mathbf{1}$ if true, otherwise 0. *The smaller* (in)stability (STA) the better. We normalize STA w.r.t. to random assignment to get $NSTA \in [0,1]$.

5.2 Datasets and Experimental Protocol

We test the techniques on thirty real classification datasets, which are diverse in size, dimensionality, number of classes and the underlying problems they represent. The characteristics are summarized in Table 1. We do not expect all of the datasets to have distinct underlying contexts and we do not know the true number of contexts. Thus, we use the *stability* measure in the evaluation to indicate whether the found contexts are persistent in the data.

In the experiments we fix the number of contexts to $k = 3$ for all the datasets (no specific reason). Feature underselection technique requires to specify the number of features, we choose $m = 5$ for all the datasets. We normalize the feature values of the input features to fall in the interval [0,1], add 1% random noise and transform the data according to its principal components. Noise does not distort class discriminatory information, neither it influences the allocation

Table 1. Datasets: **N** - size, **d** - dimensionality, **y** - number of classes

dataset	N	d	y	dataset	N	d	y	dataset	N	d	y
balance[1]	625	4	3	shuttle2[1]	14500	9	7	wis. cancer[1]	569	30	2
blood[1]	748	4	2	shuttle1[1]	43500	9	7	luxembourgh[4]	1901	31	2
mammographic[1]	961	5	2	vowels[1]	990	10	11	king-rock-pawn[1]	3196	36	2
car[1]	1728	6	4	page blocks[1]	5473	10	5	connect-4[1]	67557	42	3
elec2[6]	44235	7	2	magic[1]	19020	10	2	marketing(c)[8]	8993	48	2
chess[4]	503	8	2	marketing(d)[8]	8993	13	2	brazil[3]	50000	49	2
pima[1]	768	8	2	adult[1]	32561	14	2	spam[8]	4601	57	2
nursery[1]	12960	8	5	australian[1]	690	15	2	ozone8[1]	2534	72	2
tic-tac-toe[1]	958	9	2	vehicles[1]	846	18	4	ozone1[1]	2536	72	2
contraceptive[1]	1473	9	3	german[1]	1000	24	2	user1[2]	1500	100	2

of contexts. Noise and principal component rotation are needed to prevent ill posed covariance matrixes of some high dimensional datasets.

We empirically explore and compare five techniques: OVE, PRO, FUS, CLU and RAN. CLU is an ordinary clustering, which we use as the baseline method. Overlay (OVE), Projection (PRO) and Feature underselection (FUS) are the three context identification techniques introduced in this paper. RAN is a benchmark partitioning technique (sanity check), which assigns contexts uniformly at random. In this study we use k-means as the base clustering technique.

5.3 Results

Table 2 presents the results aggregated into three groups based on the dimensionality of the datasets: small (up to 10 features), medium (10-19 features) and large (more than 20 features) and all together. The results are plotted in Figure 5, where each dot represent one dataset. The figure shows that in terms of not capturing class labels (NMI) Overlay and Projection are doing well, while Feature underselection and the baseline Clustering are doing not that well. High NMI is consistent with higher validity, where Clustering outperforming the others, as presented in Table 2.

For all the techniques the validity deteriorates with increase in dimensionality. It can be expected, challenges of measuring distance in high dimensional space has been widely acknowledged [1]. FUS technique demonstrates the worst validity in high dimensional space. It can be explained by relatively low number of the selected features (we fixed $m = 5$).

Clustering has the best validity and stability, but captures a part of class discriminatory information, as expected, especially in low dimensional tasks. Projection has fine independence, good validity but it is rather unstable. This is mostly due to slicing of the resulting 1D projection. Likely, the resulting cut points might be not optimal and induce instability.

[1] UCI Irvine Machine Learning Repository http://archive.ics.uci.edu/ml/
[2] Katakis http://mlkd.csd.auth.gr/concept_drift.html
[3] PAKDD 2009 competition http://sede.neurotech.com.br:443/PAKDD2009/
[4] Žliobaitė collection http://sites.google.com/site/zliobaite/resources-1

Table 2. Summary of context identification results (the best in **bold**)

		CLU	OVE	PRO	FUS	RAN		CLU	OVE	PRO	FUS	RAN
	NMI	0.12	**0.02**	0.05	0.03	0		NMI	0.13	**0.02**	0.06	0.05	0
all	val.	0.12	0.18	0.18	**0.16**	1	small	val.	0.05	0.11	0.15	**0.08**	1
	stab.	0.36	**0.38**	0.59	0.53	1		stab.	0.43	**0.45**	0.63	0.49	1
	NMI	0.09	0.01	**0.01**	0.01	0		NMI	0.13	0.05	0.11	**0.03**	0
large	val.	0.19	0.26	**0.25**	0.27	1	med	val.	0.11	0.17	**0.11**	0.14	1
	sta.	0.32	**0.37**	0.50	0.67	1		sta.	0.29	**0.28**	0.68	0.38	1

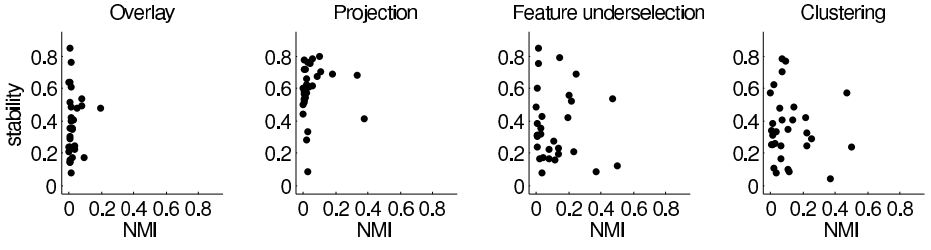


Fig. 5. Context identification results

Feature underselection has surprisingly low stability as well as mediocre validity in high dimensional tasks. This is explainable by a fixed number of features m in our experiments (for comparability across datasets). In high dimensional spaces the selected features make rather small share of all features and are thus more likely to represent noise rather than context or predictive information. Good news is that for designing individual context handling strategies that can be resolved by manipulating m value.

Overlay has good independence and stability, while not so good but acceptable validity. This is due to supervised learning procedure to assign context to unseen instances. It introduces extra uncertainty, while the other techniques can identify context for an unseen instance directly.

To sum, all three techniques avoid capturing class label information well and show similar validity; in terms of stability, Overlay technique is preferable.

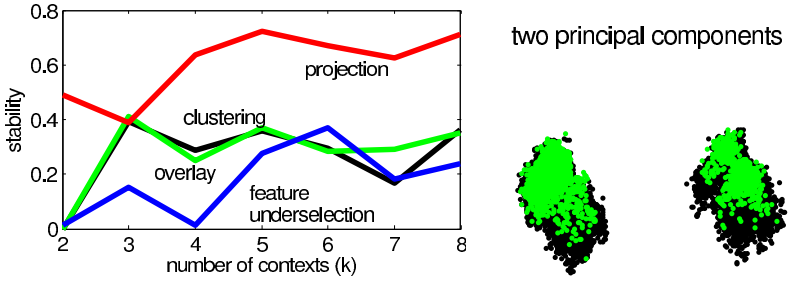


Fig. 6. Determining the number of contexts for *adult* data

6 Case Study

The following case study illustrates how context information can be used to benefit the final classification. We present it as a proof of concept rather than an attempt to select the most accurate context handling strategy. The case study focuses on determining the number of contexts (k) from the data and building one local classifier for each context.

We use *adult* dataset (alphabetically). We consider it suitable because of two reasons: the task (predicting income of a person) intuitively is context dependent and the dataset is relatively large (> 30 th. instances).

We evaluate the accuracies using six base classifiers: decision tree (CART), logistic regression, Naive Bayes, linear discriminant (LDA), neural network (with 4 hidden layers), 1-nearest neighbor (1NN), and a collection. Collection means that the most accurate base classifier is selected from a pool of all but neural network (as it performs well on its own). We run 10-fold cross validation.

We run four context identification techniques (CLU, OVE, PRO, FUS) using different number of contexts $k = 2 \ldots 8$. The resulting stabilities are presented in Figure 6. Several strategies show the best stability at two contexts, then at four and seven. This tendency is also visible from the two principal components in the same figure. We choose to analyze $k = 4$ for this case study, since it is more interesting because of a larger distinction from single context. For a full picture, we also report the ranking of the techniques at $k = 2$ and $k = 7$.

How do we know that there are variable contexts at all? It can be concluded from the stability test and the plot of principal components. If there were no distinct contexts, the stability would be bad and the data in the principal component plot would be mixed.

Set up. The simplest context handling strategy is to build one local classifier for each context. We test how it works using the context labels identified by our techniques. For comparison we include a random split into contexts (RAN) and no split into contexts (ALL), which we use as baselines. We also add to the tests an ensemble (ENS) of CLU, OVE, PRO, FUS and RAN, which makes classification decision using simple majority voting.

The testing errors are provided in Table 3. For the final evaluation, we average over the errors of different classifiers. Statistical significance is tested using a paired t-test. Symbol '$\bullet$' means the technique is significantly better than the baseline (ALL). symbol '$\circ$' means the technique is significantly worse than the baseline. Symbol '$-$' means no statistical difference.

In terms of accuracy CLU performs not bad, NMI score shows that it captures not so much class label information on this data. Interestingly, RAN sometimes outperforms ALL. It can be seen as a variant of boosting, though suffering from small training sample. OVE and FUS performs on average better than CLU, it is mainly due to bad performance of CLU on the last test (collection).

We find that an ensemble (ENS) is the best in terms of accuracy. It is supported by experiments with different number of contexts. The rankings are:

Table 3. Errors of local classifiers

	CLU	OVE	PRO	FUS	RAN	ALL	ENS
cart tree	19.62−	19.66−	19.60−	20.03−	20.44○	19.70	16.73●
log. reg.	16.51●	16.82●	17.43−	17.23●	17.45−	17.46	16.72●
n. bayes	19.28−	18.32●	19.02●	18.70●	19.31−	19.31	18.76●
LDA	21.62●	22.57●	22.30●	22.56●	23.28−	23.35	21.64●
neural n.	15.18●	15.79−	15.29●	15.83−	15.58●	15.91	15.06●
kNN	20.51−	20.62○	20.51−	20.54−	21.61○	20.50	20.47−
collection	19.86○	17.10●	18.71○	17.46−	17.52−	17.43	16.62●
mean	18.94	18.70	18.98	18.91	19.32	19.10	**18.00**

$k = 2$ ENS$\prec$PRO$\prec$FUS$\prec$OVE$\prec$ALL$\prec$CLU$\prec$RAN;
$k = 4$ ENS$\prec$OVE$\prec$FUS$\prec$CLU$\prec$PRO$\prec$ALL$\prec$RAN;
$k = 7$ ENS$\prec$PRO$\prec$ALL$\prec$OVE$\prec$FUS$\prec$RAN$\prec$CLU.

The scope of the study is to analyze context identification rather than explore context handling strategies. Thus, we explore in depth only selection strategy and do not claim that it is the best. We report it as an illustration, complementary to the proposed identification techniques. It demonstrates, how the accuracy can be improved having no domain knowledge about underlying contexts, starting from identification of the number of contexts to training the actual classifiers.

7 Conclusion

Context identification techniques can be considered as a preprocessing step in classification, aimed to improve the accuracy, as well as contribute to understanding of the data. We require the contexts to be independent from the class labels, valid (non random) and stable.

We proposed three techniques for identifying hidden contexts from the data, directed not to capture class discriminatory information. The experiments on thirty datasets indicate that all the three techniques avoid capturing class label information pretty well and show similar validity; in terms of stability Overlay technique is preferable. The case study illustrates the benefits of context identification when used with classifier selection strategy.

Our study opens a range of follow up research opportunities for context handling strategies in static and dynamic (concept drift, discrimination aware learning) settings.

References

1. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 217–235. Springer, Heidelberg (1998)
2. Brézillon, P.: Context in problem solving: a survey. Knowledge Engineering Review 14(1), 47–80 (1999)

3. Dara, R.A., Makrehchi, M., Kamel, M.S.: Filter-based data partitioning for training multiple classifier systems. IEEE Trans. on Knowledge and Data Engineering 22(4), 508–522 (2010)
4. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience Publication, Hoboken (2000)
5. Frosyniotis, D., Stafylopatis, A., Likas, A.: A divide-and-conquer method for multi-net classifiers. Pattern Analysis and Applications 6(1), 32–40 (2003)
6. Harries, M.: Splice-2 comparative evaluation: Electricity pricing. Technical report, U. New South Wales (1999)
7. Harries, M., Sammut, C., Horn, K.: Extracting hidden context. Machine Learning 32(2), 101–126 (1998)
8. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference and prediction. Springer, Heidelberg (2005)
9. Katakis, I., Tsoumakas, G., Vlahavas, I.: Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowledge Information Systems 22(3), 371–391 (2010)
10. Lim, M., Sohn, S.: Cluster-based dynamic scoring model. Expert Systems with Appl. 32(2), 427–431 (2007)
11. Liu, R., Yuan, B.: Multiple classifiers combination by clustering and selection. Information Fusion 2(3), 163–168 (2001)
12. Ren, J., Shi, X., Fan, W., Yu, P.S.: Type-independent correction of sample selection bias via structural discovery and re-balancing. In: Proc. of the SIAM Int. Conf. on Data Mining (SDM 2008), pp. 565–576 (2008)
13. Roth, V., Lange, T., Braun, M., Buhmann, J.: A resampling approach to cluster validation. In: Proc. of Int. Conf. on Computational Statistics, pp. 123–128 (2002)
14. Strang, T., Linnhoff-Popien, C.: A context modeling survey. In: Workshop on Advanced Context Modelling, Reasoning and Management at the 6th Int. Conf. on Ubiquitous Computing (UbiComp 2004) (2004)
15. Turney, P.: The identification of context-sensitive features: A formal definition of context for concept learning. In: Proc. of the ICML 1996 Workshop on Learning in Context-Sensitive Domains, pp. 53–59 (1996)
16. Turney, P.: The management of context-sensitive features: A review of strategies. In: Proc. of the ICML 1996 Workshop on Learning in Context-Sensitive Domains, pp. 60–65 (1996)
17. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning 23(1), 69–101 (1996)
18. Wu, M., Scholkopf, B.: A local learning approach for clustering. In: Advances Neural Information Processing Systems (NIPS 2006) (2006)

Cross-Lingual Sentiment Classification via Bi-view Non-negative Matrix Tri-Factorization

Junfeng Pan, Gui-Rong Xue, Yong Yu, and Yang Wang

Shanghai Jiao Tong University
800 Dongchuan Rd., Shanghai 200240, China
{panjf,grxue,yyu}@apex.sjtu.edu.cn, y_wang@sjtu.edu.cn

Abstract. Recently the sentiment classification problem interests the researchers over the world, but most sentiment corpora are in English, which limits the research progress on sentiment classification in other languages. Cross-lingual sentiment classification aims to use annotated sentiment corpora in one language (e.g. English) as training data, to predict the sentiment polarity of the data in another language (e.g. Chinese). In this paper, we design a bi-view non-negative matrix tri-factorization (BNMTF) model for the cross-lingual sentiment classification problem. We employ machine translation service so that both training and test data is able to have two representation, one in source language and the other in target language. Our BNMTF model is derived from the non-negative matrix tri-factorization models in both languages in order to make more accurate prediction. Our BNMTF model has three main advantages: (1) combining the information from two views (2) incorporating the lexical knowledge and training document label knowledge (3) adding information from test documents. Experimental results show the effectiveness of our BNMTF model, which can outperform other baseline approaches to cross-lingual sentiment classification.

Keywords: Sentiment, Cross-Lingual, Matrix Factorization.

1 Introduction

Sentiment classification is the task to predict the sentiment polarity of a given review document or a comment sentence. In recent years, sentiment classification interests more and more researchers in natural language processing and data mining fields because of rapid development of techniques and growing population of sentiment-rich resources.

But most annotated sentiment corpora are in English, and annotating the sentiment corpora in other language is difficult, time-consuming or expensive. In order to solve the lack of annotated data problem, cross-lingual sentiment classification aims to use annotated sentiment corpora in one language (e.g. English) as training data, to predict the sentiment polarity of the data in another language (e.g. Chinese). Some pilot studies projected the data in target language into source language, and then treated the problem as sentiment classification in single language. Although various projection techniques (dictionary-based, machine-translation-based, etc.) have been applied, the performance is far from satisfactory.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 289–300, 2011.

In this paper, we target at finding an effective approach for the cross-lingual sentiment classification problem. To achieve our goals, we propose a bi-view non-negative matrix tri-factorization (BNMTF) model. Our BNMTF model attacks the problem from the following directions:

- Combine the information from both the source language view and the target language view by connecting the two views through a useful and important constraint.
- Incorporate both the lexical knowledge and the document label knowledge by extended non-negative matrix factorization.
- Add the information from the test data by transductive learning setting.

The rest of the paper is organized as follows. We review some related work in Section 2. Then we give the formal definition and setting of the cross-lingual sentiment classification problem we address in this paper in Section 3. The BNMTF model is proposed in Section 4, and we also give some brief analysis on our solution. We conduct a series of experiments to evaluate the effectiveness of our proposed model in Section 5. Finally, we conclude our work in Section 6.

2 Related Work

In this section, we review the prior researches mostly related to our work. Most of the related work comes from two research areas, cross-lingual information access and sentiment classification.

Cross-lingual information access (CLIA) is concerned with technologies and applications that enable people to freely access information that is expressed in any language. A lot of previous work emphasizes on cross-lingual text classification and clustering. [2] studied English-Spanish cross-language classification problem in the poly-lingual and cross-lingual cases. [11] classified Czech documents using the classifier built on English training data by a probabilistic English-Czech dictionary. [9] developed a novel method known as the information bottleneck technique for Chinese web pages classification using the English web pages as training data. [17] proposed a framework to cross-lingual query classification which extended the cross-lingual classification problem into short text case. [18] incorporated document similarity propagation method with the spectral clustering algorithm to cluster the news in different languages.

In this paper, we focus on cross-lingual text classification and we also employ the machine translate system to help us project the data from one language into another language.

Sentiment classification aims to predict the sentiment polarity of text data. The approaches to solve the problem can be generally categorized into lexicon-based and corpus-based. Lexicon-based approaches measure the sentiment of the text based on sentiment lexicons. [14] proposed the semantic oriented method to predict the sentiment orientation of a review while [6] built three models to assign sentence sentiment. Corpus-based approaches classify the sentiment of a given sentence or document by the classifier built using labeled sentiment data. Since the work of [13], various corpus-based methods have proposed.

An important issue in sentiment classification is the domain dependency problem, so a lot of efforts on domain adaptation have been made. [3] proposed the structural correspondence learning (SCL) algorithm while [12] proposed the spectral feature alignment (SFA) algorithm for the cross-domain sentiment classification problem. And we can also treat the cross-lingual adaptation as a special case of domain adaptation if we consider different languages as different domains. The cross-lingual sentiment classification problem has also been studied by some previous work. [10] used cross-lingual projection through dictionaries or parallel corpora to judge the document subjectivity, while [1] and [15] employed the machine translation systems to bridge the data in different languages. [16] proposed to use the co-training approach to address the problem.

In this paper, we try to combine the lexicon-based and corpus-based approaches like previous work on non-negative matrix tri-factorization [8]. In fact, the NMTF model can be treated as the basic component of our models. We refine the model to satisfy the cross-lingual condition in the problem we address in this paper. And Like [16], we also view the classification problem in two independent views, the source language view and the target language view. But we have two main difference between the standard co-training approach. One is that our model incorporate the lexical knowledge, and the other is that we combine the two views in the matrix factorization process so that we only need to train the BNMTF classifier once while a series of classifiers need to be trained if the standard co-training approach is applied.

3 Problem Setting

Before giving a formal definition of the problem we address in this paper, we first present some definitions.

Definition 1. *(Vocabulary)* V_s *denotes the vocabulary in source language, while* V_t *denotes the vocabulary in target language. Furthermore,* $|V_s| = m_s$ *and* $|V_t| = m_t$.

In this paper, we use English as source language and Chinese as target language.

Definition 2. *(Sentiment Lexicon)* $W_s^+ = \{w_{s\,1}^+, w_{s\,2}^+, ...\}$ *denotes the positive lexicon in source language, while* $W_s^- = \{w_{s\,1}^-, w_{s\,2}^-, ...\}$ *denotes the negative lexicon in source language. Similarly, we can define* W_t^+ *and* W_t^-.

Sentiment lexicons are very useful and important to the lexicon-based approaches for sentiment classification.

Definition 3. *(Document Set)* $D_s = \{d_{s1}, d_{s2}, ..., d_{sn_s}\}$ *denotes the document set in source language. We have* $d_{si} \in V_s^*$. *Similarly, we can define* D_t.

Definition 4. *(Label)* $Y_s = \{y_{s1}, y_{s2}, ..., y_{sn_s}\}$ *denotes the label set for the document set in source language.* $y_{si} = +1$ *if the overall sentiment expressed in* d_{si} *is positive, while* $y_{si} = -1$ *if the overall sentiment expressed in* d_{si} *is negative. Similarly, we can define* Y_t.

Please notice that in the problem we address in this paper, Y_s is given, but the Y_t is unknown and needs us to predict.

As sentiment lexicons to the lexicon-based approaches, labeled documents are very useful and important to the corpora-based approaches for sentiment classification.

Definition 5. *(Translator)* $T_{src \to tar} : V_s^* \to V_t^*$ *denotes a translator from the source language to the target language. And* $d_{s_i}^t$ *denotes* $T_{src \to tar}(d_{s_i})$ *while* D_s^t *denotes* $T_{src \to tar}(D_s)$*. Similarly, we can define* $T_{tar \to src}$*,* $d_{t_i}^s$ *and* D_t^s*.*

In practice, translators can be dictionaries, machine translation systems, or other projections from one language to another language. Through the translators, each document is able to have two views (one in source language and the other in target language) despite its original language.

Based on the definitions described above, now we can define the problem we try to address in this paper as follows:

Definition 6. *(Cross-Lingual Sentiment Classification) Given* V_s*,* V_t*,* W_s^+*,* W_s^-*,* W_t^+*,* W_t^-*,* D_s*,* D_t*,* Y_s*,* $T_{src \to tar}$ *and* $T_{tar \to src}$*, we need to predict* Y_t*.*

In order to solve this problem, we try to achieve the following subgoals:

- Combine the information from both the source language view and the target language view because both views are helpful and able to contribute to the prediction accuracy.
- Incorporate both the lexical knowledge (lexicon-based approach) and the document label knowledge (corpora-based approach) into the model.
- Furthermore, adding information from the test unlabeled data to the model can help reduce the distribution divergence between the training data and test data.

4 Bi-view Non-negative Matrix Tri-Factorization

In this section, we describe our basic idea of our bi-view non-negative matrix factorization (BNMTF) model, and then give the mathematical formulation. We also analyze our solution briefly in this section.

4.1 Basic Idea

Firstly, let's consider the problem in the source language side. We build a term-document matrix X_s using V_s, D_s and D_t^s. Here machine translation service is employed so that all the documents(including training and test) have representations in both source language and target language. And in this paper, we use both the training documents and the test documents together to build the term-document matrix in order to add knowledge from the unlabeled data to the model, which means the models we proposed here is transductive (i.e., we use test data without label in training phrase).

And then similar to [8], we can set up a non-negative matrix tri-factorization (NMTF) problem as equation (1).

$$\min_{F_s \geqslant 0, G_s \geqslant 0, S_s \geqslant 0} \| X_s - F_s S_s G_s^{\mathrm{T}} \|^2 + \alpha_s \mathrm{Tr}[(F_s - F_{0s})^{\mathrm{T}} C_{1s} (F_s - F_{0s})] + \beta_s \mathrm{Tr}[(G_s - G_{0s})^{\mathrm{T}} C_{2s} (G_s - G_{0s})] \quad (1)$$
$$\alpha_s > 0, \beta_s > 0, \sum_{ij} X_{si,j} = 1, \sum_k F_{sik} = 1, \sum_k G_{sjk} = 1, \sum_k S_{skk} = 1$$

We normalize these matrices so that we can explain the equation using the concepts in probabilistic latent semantic indexing (PLSI) model [5]. X_s, F_s, G_s and S_s can be treat as the joint distribution between words and documents, the word class-conditional probability, the document class-conditional probability and the class probability distribution.

According to [8], the lexical knowledge and training document label knowledge can be incorporated. Here we incorporate them by the following ways:

- (Lexical Knowledge) We set $(F_{0s})_{i1} = p$ ($p \approx 1$) and $(F_{0s})_{i2} = 1 - p$ if the i-th word is positive, while $(F_{0s})_{i2} = p$ and $(F_{0s})_{i1} = 1 - p$ if the i-th word is negative. And we set $(C_{1s})_{ii} = 1$ if we know the lexical knowledge of the i-th word and $(C_{1s})_{ii} = 0$ otherwise.
- (Training Document Label Knowledge) We set $(G_{0s})_{j1} = q$ ($q \approx 1$) and $(G_{0s})_{j2} = 1 - q$ if the j-th document is positive, while $(G_{0s})_{j2} = q$ and $(G_{0s})_{j1} = 1 - q$ if the j-th document is negative. And we set $(C_{2s})_{jj} = 1$ if we know the label of the j-th document and $(C_{1s})_{jj} = 0$ otherwise.

For the test documents, it is better if good prior can be given. In [8], the authors use the K-means clustering results for initialization. Here we proposed an alternative way. We use the training data to build a traditional classifier (e.g. maximum entropy classifier, support vector machine, etc.), and then using the classification results on the test data as prior knowledge. In our model, we also put this prior knowledge into G_{0s}.

Now we have given the details of our model in source language view, while in the target language view everything is almost the same. Similar to equation (1), we can set up another non-negative matrix tri-factorization problem as equation (2) using the data in target language view.

$$\min_{F_t \geqslant 0, G_t \geqslant 0, S_t \geqslant 0} \| X_t - F_t S_t G_t^{\mathrm{T}} \|^2 + \alpha_t \mathrm{Tr}[(F_t - F_{0t})^{\mathrm{T}} C_{1t} (F_t - F_{0t})] + \beta_t \mathrm{Tr}[(G_t - G_{0t})^{\mathrm{T}} C_{2t} (G_t - G_{0t})] \quad (2)$$
$$\alpha_t > 0, \beta_t > 0, \sum_{ij} X_{ti,j} = 1, \sum_k F_{tik} = 1, \sum_k G_{tjk} = 1, \sum_k S_{tkk} = 1$$

The corresponding matrices can be set in the same way as in source language view.

After we set up the two NMTF problems, we see them as two views of our model and connect them in order to combine the information from the two views. Equation (3) is a simple constraint which gives a useful connection: The sentiment of a document is the same (at least close to each other) in the two views.

$$G_s \approx G_t \quad (3)$$

The schema of our basic idea is shown in Figure 1. We build two term-document matrices, one according to representation of all the documents(both training and test) in the source language and the other in the target language. The transductive learning setting ensure the information from test data included. And then we factorize the two matrices into document and word space, the lexical knowledge and training document label knowledge is incorporated in the factorization process. The factorization process is also under a useful constraint so that we can combine the information from two views.

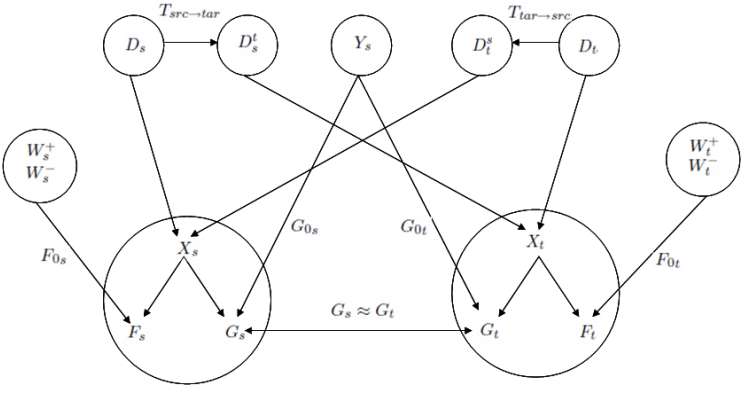


Fig. 1. The Schema of Our Basic Idea

4.2 Mathematical Formulation and Brief Analysis

In this subsection, we propose our BNMTF model in mathematical formulation and a iterative solution to the BNMTF with some brief analysis.

If we put equation (1) and (2) together with a bi-view coefficient γ and let $G_s = G_t = G$, we get equation (4). It is a good mathematical representation of our basic idea in last subsection.

$$\begin{aligned}
&\min_{F_s\geqslant 0,S_s\geqslant 0,F_t\geqslant 0,S_t\geqslant 0,G\geqslant 0} \gamma\{\| X_s - F_s S_s G^{\mathrm{T}} \|^2 \\
&+\alpha_s \mathrm{Tr}[(F_s - F_{0s})^{\mathrm{T}} C_{1s}(F_s - F_{0s})] + \beta_s \mathrm{Tr}[(G - G_{0s})^{\mathrm{T}} C_{2s}(G - G_{0s})]\} \\
&+(1-\gamma)\{\| X_t - F_t S_t G^{\mathrm{T}} \|^2 \\
&+\alpha_t \mathrm{Tr}[(F_t - F_{0t})^{\mathrm{T}} C_{1t}(F_t - F_{0t})] + \beta_t \mathrm{Tr}[(G - G_{0t})^{\mathrm{T}} C_{2t}(G - G_{0t})]\} \qquad (4) \\
&\alpha_s > 0, \beta_s > 0, \sum_{ij} X_{si,j} = 1, \sum_k F_{sik} = 1, \sum_k S_{skk} = 1 \\
&\alpha_t > 0, \beta_t > 0, \sum_{ij} X_{ti,j} = 1, \sum_k F_{tik} = 1, \sum_k S_{tkk} = 1, \sum_k G_{jk} = 1
\end{aligned}$$

$\gamma \in [0, 1]$ is the bi-view coefficient which indicate how confident we are to each view.

To solve equation (4), we design a iterative procedure which is list as follows[1].

- $r = s/t$
- Iteration:

$$G_{jk} \leftarrow G_{jk} \frac{[\gamma(X_s^{\mathrm{T}} F_s S_s + \beta_s C_{2s} G_{0s}) + (1-\gamma)(X_t^{\mathrm{T}} F_t S_t + \beta_t C_{2t} G_{0t})]_{jk}}{[\gamma(GG^{\mathrm{T}} X_s^{\mathrm{T}} F_s S_s + \beta_s C_{2s} G) + (1-\gamma)(GG^{\mathrm{T}} X_t^{\mathrm{T}} F_t S_t + \beta_t C_{2t} G)]_{jk}} \tag{5}$$

$$F_{rik} \leftarrow F_{rik} \frac{(X_r G_r S_r^{\mathrm{T}} + \alpha_r C_{1r} F_{0r})_{ik}}{(F_r F_r^{\mathrm{T}} X_r G_r S_r^{\mathrm{T}} + \alpha_r C_{1r} F_r)_{ik}} \tag{6}$$

$$S_{rik} \leftarrow S_{rik} \frac{(F_r^{\mathrm{T}} X_r G_r)_{ik}}{(F_r^{\mathrm{T}} F_r S_r G_r^{\mathrm{T}} G_r)_{ik}} \tag{7}$$

- Initialization:

$$G \leftarrow \gamma G_{0s} + (1-\gamma) G_{0t} \tag{8}$$

$$F_r \leftarrow F_{0r} \tag{9}$$

$$S_r \leftarrow F_r^{\mathrm{T}} X_r G \tag{10}$$

Then, we briefly analyze the correctness[2], convergence[3] and computational complexity of our updating rules.

For proving the correctness, we need just calculate the gradient of the objective function, and then test the KKT condition like [4].

For proving the convergence, we need just use auxiliary function approach like [7]. And empirically, we need only a few iterations before practical convergence (see in Subsection 5.4).

Last but not the least, let us consider about computational complexity. The same as NMTF model, our BNMTF model scales linearly with the dimensions and density of the term-document matrices. For one iteration, it takes $O(k^2(m_s+m_t+n_s+n_t)+kz)$ time, while $k = 2$ and $z \ll (m_s+m_t)(n_s+n_t)$ is the number of non-zero entries in the two term-document matrices here.

5 Experiments

In this section, we describe our experiments to show the effectiveness of our proposed bi-view non-negative matrix factorization (BNMTF) model for cross-lingual sentiment classification.

[1] We normalize the matrices after each iteration.
[2] The algorithm converges to a local optima.
[3] The algorithm converges in finite steps.

5.1 Datasets

We use three review datasets in different domains (MOVIE/BOOK/MUSIC) in the experiments. On the English side, we use movie reivews from the IMDB archives used in [13], book and music reviews from Amazon.com used in [3], while on the Chinese side, we collect movie, book and music reviews from a popular Chinese social web, douban[4]. We treat the English reviews as training data, and use the Chinese reviews for test. Furthermore, to evaluate our algorithm, we label the Chinese reviews according to the users' ratings.

For all the datasets, all the reviews are translated into another language by Google Translate Service[5]. The data in English view is tokenized and lowercased by some perl scripts[6], while the data in Chinese view is segmented by the Stanford Chinese Word Segmenter[7]. At last stopwords are removed.

The summary of the review datasets is shown in Table 1.

Table 1. Summary of the review datasets

Dataset	Language	#Positive	#Negative	#Terms in Chinese	#Terms in English
MOVIE	Eng.	1,000	1,000	84,530	62,091
	Chn.	1,000	1,000		
BOOK	Eng.	1,000	1,000	54,295	35,788
	Chn.	1,000	1,000		
MUSIC	Eng.	1,000	1,000	49,085	30,730
	Chn.	1,000	1,000		

And we also use four sentiment lexicons from HowNet Knowledge Database[8]. Totally, there are 3,730 positive words and 3,116 negative words in Chinese, with 3,594 positive words and 3,563 negative words in English.

5.2 Baselines

In the experiments, we use the following baseline methods to compare with our BNMTF model described in Subsection 4.2. For the first three baselines, only training data is used in training phrase, while for others, all the data is used (except the labels of test data).

- MaxEnt(CHN): It applies maximum entropy classifier[9] with only Chinese terms.
- MaxEnt(ENG): It applies maximum entropy classifier with only English terms.
- MaxEnt(CE): It averages the results of MaxEnt(CHN) and MaxEnt(ENG).

[4] http://www.douban.com
[5] http://translate.google.com/
[6] http://www.statmt.org/wmt08/scripts.tgz
[7] http://nlp.stanford.edu/software/segmenter.shtml
[8] http://www.keenage.com/
[9] We use Maximum Entropy Modeling Toolkit for Python and C++ by Le Zhang, http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html

- CoTrain: It is like the co-training approach for cross-lingual sentiment classification in [16]. The only difference is here we use the test data as unlabeled auxiliary data.
- NMTF(CHN): It solves equation(2). The prediction of MaxEnt(CHN) on test data is used as prior knowledge.
- NMTF(ENG): It solves equation(1). The prediction of MaxEnt(ENG) on test data is used as prior knowledge.
- NMTF(CE): It averages the results of NMTF(CHN) and NMTF(ENG).

We also calculate the 10-fold for every dataset by 10-fold cross validation on the test data using maximum entropy classifier. It is an approximation of the upper bound of classification results.

A quick word about parameter setting. According to [8], we set $\alpha_{src} = \alpha_{tar} = 0.5$ and $\beta_{src} = \beta_{tar} = 1$. And we set $\gamma = 0.5$ and $I = 100$ (the number of iterations for non-negative matrix tri-factorization) for the overall comparison.

5.3 Overall Comparison Results

We use accuracy to evaluate the classification result.

$$accuracy = \frac{\#\{samples\_classified\_correctedly\}}{\#\{all\_samples\}} \tag{11}$$

Figure 2 shows the overall comparison results. From the figure, we can observe that our BNMTF model outperforms other baseline methods significantly.

The first three baseline methods do not use the unlabeled test data in training phrase, so they can not perform as good as other methods which are benefited by the information from the unlabeled test data. Among all the baselines, the accuracy of CoTrain is almost the closest to that of our BNMTF model because these two methods combine the information both Chinese view and English view in the whole process, where others are trained in only one view or combine the two views in the last time. We can also see that generally NMTF models are better than corresponding MaxEnt models because of the sentiment lexical knowledge. It indicates that both sentiment lexicons and labeled training documents is useful and important to sentiment classification task. It is also the main reason why our BNMTF outperform the strongest baseline, CoTrain. We also note that on MUSIC, our BNMTF model even outperform the uppperbound. It is reasonable because our BNMTF model contains the sentiment lexical knowledge which is not used when we calculate the 10-fold.

5.4 Influence of Parameters

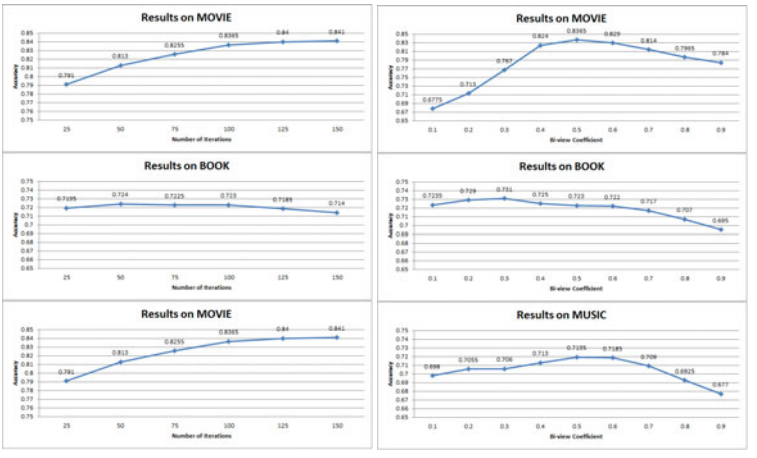
We test two main parameters of our BNMTF model, the number of iterations (I) and the bi-view coefficient (γ).

Figure 3(a) shows the influences of number of iterations to the classification performance. In this experiment, we fix $\gamma = 0.5$ and change the value of I from 25 to 150 with step length 25. We can see that after 100 iterations, the classification accuracy become robust. On MOVIE and MUSIC, the accuracy increases after 100 iterations but the speed is slower. On BOOK, the accuracy decreases after 100 iterations because the

Fig. 2. The Overall Comparison Results

noise of the data, but the results are still acceptable. The experiment results show that our algorithms converge fast in practice, which implies the scalability of our approach.

Figure 3(b) shows the influences of bi-view coefficient to the classification performance. In this experiment, we fix $I = 100$ and change the value of γ from 0.1 to 0.9 with step length 0.1. Generally speaking, the classification accuracy is higher when γ is closer to 0.5. The experiment results indicate that both Chinese view and English view are beneficial to classification accuracy. And our BNMTF model, which combines the information from the two views, is able to gain the benefits from both views. Although the Chinese view on BOOK is a little poor so that we get the best results on BOOK when $\gamma = 0.3$ not $\gamma = 0.5$, we can also see the benefit of the Chinese view. Furthermore, this situation implies that we can estimate the parameter via a validation set instead of setting it manually.



(a) Number of Iterations (b) Bi-view Coefficient

Fig. 3. The Influence of Parameters

6 Conclusion and Future Work

In this paper, we propose a bi-view non-negative matrix tri-factorization (BNMTF) model for the cross-lingual sentiment classification problem. Our model have three main advantages, (1) combining the information from two views by a useful and important constraint, (2) incorporating the lexical knowledge and training document label knowledge by the extended NMTF and (3) adding information from test documents by the transductive learning setting. Our experiments on cross-lingual sentiment classification in three different domains demonstrate the effectiveness of our proposed approach.

In the future, we are planning to extend our BNMTF model from transductive learning setting into inductive learning setting. To do this, we need just use other unlabeled data[10] instead of the test data in training phrase, and then predict the test data in the non-negative matrix tri-factorization framework. In additional, currently we estimate all the parameters manually, while it seems more reasonable if we can estimate parameters via a validation set. Finally, we are also planning to extend our proposed BNMTF models to solve more general cross-lingual classification and other cross-lingual information access problems.

Acknowledgments

This work is supported by National Natural Science Foundation of China(No. 60873211). We also thank the anonymous reviewers for their elaborate and helpful comments.

[10] The additional unlabeled data should be under the same distribution as the test data.

References

1. Banea, C., Mihalcea, R., Wiebe, J., Hassan, S.: Multilingual subjectivity analysis using machine translation. In: EMNLP, pp. 127–135. Association for Computational Linguistics, Morristown (2008)
2. Bel, N., Koster, C.H.A., Villegas, M.: Cross-lingual text categorization. In: Koch, T., Sølvberg, I.T. (eds.) ECDL 2003. LNCS, vol. 2769, pp. 126–139. Springer, Heidelberg (2003)
3. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL, pp. 440–447. Association for Computational Linguistics, Morristown (2007)
4. Ding, C., Li, T., Peng, W., Park, H.: Orthogonal nonnegative matrix tri-factorizations for clustering. In: KDD, pp. 126–135. ACM, New York (2006)
5. Hofmann, T.: Probabilistic latent semantic indexing. In: SIGIR, pp. 50–57. ACM, New York (1999)
6. Kim, S.M., Hovy, E.: Determining the sentiment of opinions. In: COLING, p. 1367. Association for Computational Linguistics, Morristown (2004)
7. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS, The MIT Press, Cambridge (2000)
8. Li, T., Zhang, Y., Sindhwani, V.: A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In: ACL, pp. 244–252. Association for Computational Linguistics, Morristown (2009)
9. Ling, X., Xue, G.R., Dai, W., Jiang, Y., Yang, Q., Yu, Y.: Can Chinese web pages be classified with English data source? In: WWW, pp. 969–978. ACM, New York (2008)
10. Mihalcea, R., Banea, C., Wiebe, J.: Learning multilingual subjective language via cross-lingual projections. In: ACL, pp. 976–983. Association for Computational Linguistics, Morristown (2007)
11. Olsson, J.S., Oard, D.W., Hajič, J.: Cross-language text classification. In: SIGIR, pp. 645–646. ACM, New York (2005)
12. Pan, S.J., Ni, X., Sun, J.t., Yang, Q., Chen, Z.: Cross-domain sentiment classification via spectral feature alignment. In: WWW, pp. 751–760. ACM, New York (2010)
13. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: EMNLP, pp. 79–86. Association for Computational Linguistics, Morristown (2002)
14. Turney, P.D.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: ACL, pp. 417–424. Association for Computational Linguistics, Morristown (2002)
15. Wan, X.: Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In: EMNLP, pp. 553–561. Association for Computational Linguistics, Morristown (2008)
16. Wan, X.: Co-Training for cross-lingual sentiment classification. In: ACL, pp. 235–243. Association for Computational Linguistics, Morristown (2009)
17. Wang, X., Broder, A., Gabrilovich, E., Josifovski, V., Pang, B.: Cross-language query classification using web search for exogenous knowledge. In: WSDM, pp. 74–83. ACM, New York (2009)
18. Yogatama, D., Tanaka-Ishii, K.: Multilingual spectral clustering using document similarity propagation. In: EMNLP, pp. 871–879. Association for Computational Linguistics, Morristown (2009)

A Sequential Dynamic Multi-class Model and Recursive Filtering by Variational Bayesian Methods

Xiangyun Qing and Xingyu Wang

School of Information Science and Engineering,
East China University of Science and Technology, China
{xytsing,xywang}@ecust.edu.cn

Abstract. Adaptive classification evolving over time is an important learning task that arises in many applications. In this paper, a sequential dynamic multi-class model (SDMM) is proposed for representing the multi-class adaptive learning task, which is based on the polychotomous response model and dynamic logistic regression. Multiple state chains in the SDMM are coupled by the observable labels and feature vectors. Each state chain is modeled as a first-order Markov process with time-varying covariance parameters for characterizing the non-stationary generating process of sequential labels. Augmented auxiliary variables are introduced for developing efficient inference procedures according to the popular data augmentation strategy. Variational Bayesian methods are applied to estimate the dynamic state variables and augmented auxiliary variables recursively. According to the results of recursive filtering procedures using mean-field approximation forms, one-step-ahead predicted probabilities are calculated by marginalizing the state variables. Experiment results based on both synthetic and real data show that the proposed model significantly outperforms the non-sequential static methods for the multi-class adaptive learning problems with missing labels. Encouraging results have been obtained by comprising well-known multi-class data stream algorithms.

1 Introduction

The growing interest in adaptive learning has been received from various challenging learning tasks in non-stationary environments such as non-invasive brain-computer interface (BCI), intrusion detection and fault detection. For example, online adaptation of the classifier is an important requirement for BCI as brain activity changes naturally over time [1]. In particular, qualitative and quantitative evidence indicating non-stationary in the BCI classification problem has been provided by [2]. In these real-world learning tasks, online learning approaches should be temporally adaptive when underlying concept of observable data changes over time. However, traditional static learning approaches do not care about the data generating process and ignore the dynamical characteristics of a set of observations.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 301–312, 2011.

Over the past ten years, researchers have explored several approaches to the adaptive learning. One approach uses particle filters [3], whose performance is demonstrated on the problem of fault detection of dynamical operated marine diesel engines. A non-stationary logistic regression model presented in [4], in which weights of the model evolve dynamically, uses the usual logistic function as the classifier. The extended Kalman filter is employed to update the weights in an online manner and the time-varying state noise variance parameters are estimated by a line search method. Successive research work build on the model. Sykacek et al. use the variational Kalman filtering as an inference method for sequential classification in BCI [5]. A windowed Kalman filter procedure is implied to compute a bound of the log evidence. Recently, two different approaches, i.e., dynamic logistic regression using nonlinear Kalman filters and a dynamic generalized linear model, are adopted to the sequential classification [6]. In the more closely related work, sequential Monte Carlo sampling and the extended Kalman filter have been utilized for solving the same problem [7] [8] [9]. Research also developed almost parallel in several different communities, such as data mining and statistics [10] [11]. An efficient and effective approach to handle the problem of concept change in data streams is called concept-adapting very fast decision tree (CVFDT) [12]. CVFDT builds the decision trees in an incremental manner and detects the changing data concept from a sliding window of a fixed size.

However, much of previous research efforts only focused on two-class learning problems and they did not provide explicit representations for multi-class learning problems. Traditional approaches for the multi-class problems by combining a set of two-class problems such as 1-vs.-all and 1-vs.-1, have some drawbacks. The approaches ignore the relations between the classes and have computational disadvantages. Although extensions to multi-class had been suggested by introducing multinomial distributions with time-varying parameters, corresponding inference algorithms can not be derived straightforwardly from that of two-class learning problems. Furthermore, the extensions substantially complicate the inference procedures, leading to different algorithms for statistical inference. This motivates us to investigate the multi-class learning problems. In this study, one of major contributions is that we provide a sequential dynamic multi-class model (SDMM) in conjunction with the polychotomous response model and dynamic logistic regression. Latent variables of the model follow a multivariate random walk. In a novel contribution, we employ the variational Bayesian methods in a recursive manner for estimating the dynamic latent variables from data feature vectors and labels. Approximate inference in the learning procedure is formed on each time step separately. Our proposed inference method is inspired by the recursive noise adaptive Kalman filtering where on each step the state of discrete-time linear state space model is estimated with Kalman filter [13].

The rest of this article is structured as follows. In Section 2, SDMM model is given. In Section 3, the variational Bayesian approach is introduced to approximate the posterior distributions of the latent variables in a recursive filtering manner. A learning algorithm is proposed to estimate parameters. Section 4 provides a calculation approach for one-step-ahead predictive distribution.

After presenting some experimental results in Section 5, conclusions and future research are given in Section 6.

2 Sequential Dynamic Multi-class Model

Let $x_1, ..., x_t, ..., x_T$ denote an observed input stream at time $t = 1, 2, ..., T$, labeled in K classes. In this research we are concerned with developing a sequential classifier for multi-class learning problems where K is greater than two. The data points arrive one x_t at a time t. A correct label $Z_t \in \{1, ..., K\}$ is revealed after the one-step-ahead prediction is done by the classifier, but before the next data point x_{t+1} is observed. The classifier is updated in an online manner by using the data observed at time t. Considering a given input feature vector h_t at time t , a sequential polychotomous response model to the dynamic multi-class learning has the following form:

$$w_{t,k} = w_{t-1,k} + v_{t,k} \tag{1}$$

$$y_{t,k} \sim N(h_t^T w_{t,k}, 1) \tag{2}$$

$$Z_t = j, \quad if \quad y_{t,j} = \max_{1 \le k \le K} \{y_{t,k}\} \tag{3}$$

where $v_{t,k} \sim N(0, q_{t,k} I)$ is the Gaussian process noise whose covariance is constrained to a scaled identity matrix $q_{t,k} I$. Compared to the polychotomous model proposed by [14], the d-dimensional state variable $w_{t,k}$ using a random-walk model is time-evolving to capture the non-stationary. Moreover, adopting the form of the random-walk model for state variables means that our model is best suitable for sequential classification in the presence of steady changes. Continuous auxiliary variables $y_{t,k}$ are introduced to augment multinomial probit regression model. As detailed in [15], the multinomial probit takes the following form by explicitly marginalizing the auxiliary variables:

$$\begin{aligned} p(Z_t = i|\mathbf{w}_t) &= \int \delta(y_{t,i} > y_{t,k}, \forall k \neq i) \prod_{j=1}^{K} p(y_{t,j}|w_{t,j}) d\mathbf{y} \\ &= E_{p(u)}\{\prod_{j \neq i} \Phi(u + h_t^T(w_{t,i} - w_{t,j}))\} \end{aligned} \tag{4}$$

where the random variable $u \sim N(0, 1)$ is standardized normal, $\Phi()$is the standardized normal Cumulative Distribution Function (CDF).

Fig.1 shows a directed acyclic graph for the hierarchical generating process. The next task is to estimate the state variable $w_{t,k}$ according to the observable labels and feature vectors by Bayesian filtering methods.

3 Recursive Filtering by Variational Bayes

3.1 Variational Bayes Approximation

To infer the state variables of the nonlinear state space model above, Bayes's rule is utilized for the recursive evaluation of the filtering distribution $p(w_{t,k}|Z_{1:t}, h_{1:t})$.

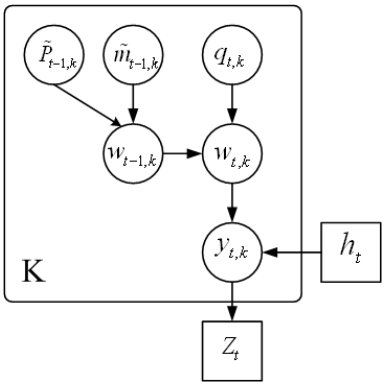


Fig. 1. Graphical model for the generating process of the observed multi-class labels given the feature vectors at time t

However, owing to the non-linear and non-Gaussian of observation function, approximation methods are required to simplify the estimation procedure. We use Variational Bayes (VB) approximation method to infer posterior distributions and variational EM to fit the hyperparameters [16].

Assume that the VB approximation has been applied at the previous time step $t-1$, i.e., the conditional distribution for $w_{t-1,k}$ given the labels $Z_{1:t-1}$ and the feature vectors $h_{1:t-1}$ takes the form as follows:

$$p(w_{t-1,k}|Z_{1:t-1}, h_{1:t-1}) = N(\tilde{m}_{t-1,k}, \tilde{P}_{t-1,k}) \tag{5}$$

Since the state process is linear, the prediction distribution is:

$$p(w_{t,k}|Z_{1:t-1}, h_{1:t-1}) = N(m^-_{t,k}, P^-_{t,k}) \tag{6}$$

where $m^-_{t,k} = \tilde{m}_{t-1,k}$ and $P^-_{t,k} = \tilde{P}_{t-1,k} + q_{t,k}I$.

Next, a mean-field approximation for the updated posterior distribution is adopted as follows:

$$p(\mathrm{w}_t, \mathrm{y}_t|Z_{1:t}, h_{1:t}) \approx \prod_{k=1}^{K} Q(w_{t,k})Q(y_{t,k}) \tag{7}$$

Following the VB methodology, a lower bound $F(Q(\mathrm{w}_t, \mathrm{y}_t), \theta_t)$ of the logarithmic marginal likelihood $\log p(Z_t, h_t|\theta_t)$ is:

$$F(Q(\mathrm{w}_t, \mathrm{y}_t), \theta_t) = \int \prod_{k=1}^{K} Q(y_{t,k})Q(w_{t,k}) \log \left[\left\{ \sum_{i=1}^{K} \delta(y_{t,i} > y_{t,k}, \forall k \neq i)\delta(Z_t = i) \right\} \prod_{k=1}^{K} p(y_{t,k}|w_{t,k})p(w_{t,k}|m^-_{t,k}, P^-_{t,k}) \right] d\mathrm{y}\, d\mathrm{w} + \sum_{k=1}^{K} H(Q(y_{t,k})) + \sum_{k=1}^{K} H(Q(w_{t,k})) \tag{8}$$

Maximizing the lower bound (8) and applying the Euler equation and constraints of Lagrange multiplier type, the solution for each $Q(w_{t,k})$ and $Q(y_{t,k})$can be derived as follows:

$$Q(w_{t,k}) = N(\tilde{m}_{t,k}, \tilde{P}_{t,k}) \quad for \quad k = 1, ..., K \tag{9}$$

$$Q(\mathrm{y}_t) = Z^{-1} \prod_k N(y_{t,k}|h_t^T \tilde{m}_{t,k}, 1) \tag{10}$$

where:

$$\begin{aligned} Z &= \int_{-\infty}^{+\infty} N(y_{t,i}|h_t^T \tilde{m}_{t,i}, 1) \prod_{j \neq i} \int_{-\infty}^{y_{t,i}} N(y_{t,j}|h_t^T \tilde{m}_{t,j}, 1) dy_{t,i} dy_{t,j} \\ &= E_{p(u)} \left\{ \prod_{j \neq i} \Phi[u + h_t^T (\tilde{m}_{t,i} - \tilde{m}_{t,j})] \right\} \end{aligned}$$

with assuming that $Z_t = i$ the i^{th} dimension of y_t is always the largest [15].

The parameters of the distributions are the following:

$$\tilde{m}_{t,k} = S_{t,k}(m_{t,k}^- + \tilde{y}_{t,k} P_{t,k}^- h_t) \tag{11}$$

$$\tilde{P}_{t,k} = S_{t,k} P_{t,k}^- \tag{12}$$

$$S_{t,k} = I - \frac{P_{t,k}^- h_t h_t^T}{1 + h_t^T P_{t,k}^- h_t} \tag{13}$$

for $Z_t = i$:

$$\tilde{y}_{t,i} = h_t^T \tilde{m}_{t,i} - \left[\sum_{j \neq i} (\tilde{y}_{t,j} - h_t^T \tilde{m}_{t,j}) \right] \tag{14}$$

for all $k \neq i$:

$$\tilde{y}_{t,k} = h_t^T \tilde{m}_{t,k} - \frac{E_{p(u)} \left\{ N(u|h_t^T (\tilde{m}_{t,k} - \tilde{m}_{t,i}), 1) \Phi_u^{t,k,i} \right\}}{E_{p(u)} \left\{ \Phi(u + h_t^T (\tilde{m}_{t,k} - \tilde{m}_{t,i}), 1) \Phi_u^{t,k,i} \right\}} \tag{15}$$

where $\Phi_u^{t,k,i} = \prod_{j \neq i,k} \Phi \left[u + h_t^T (\tilde{m}_{t,i} - \tilde{m}_{t,j}) \right]$. Unlike the parameters of $Q(w_{t,k})$ with determination solutions, that of $Q(y_{t,k})$ are obtained by importance sampling method.

Finally, we provide the expression of the variational lower bound by using (8,11-15):

$$\begin{aligned} F(Q(w_t, y_t), \theta_t) = \log Z_t - \left(\tfrac{K-1}{2}\right) \log 2\pi + \tfrac{1}{2} \sum_{k=1}^{K} \log \left| \tilde{P}_{t,k} (P_{t,k}^-)^{-1} \right| \\ + \tfrac{1}{2} \sum_{k=1}^{K} tr\{ I - [\tilde{P}_{t,k} + (\tilde{m}_{t,k} - m_{t,k}^-)(\tilde{m}_{t,k} - m_{t,k}^-)^T] (P_{t,k}^-)^{-1} \} \end{aligned} \tag{16}$$

The value of the bound can be employed to set a convergence criterion for iteratively updating the parameters of $Q(w_{t,k})$ and $Q(y_{t,k})$ using (11) to (15).

The calculation as described above is called E-step of variational Bayesian expectational maximizing algorithm (VB-EM), whose purpose is to estimate the hidden states. The M-step of VB-EM is to estimate the state evolution noise variance $q_{t,k}$ using the states estimated in the previous E-step. Ideally, all $q_{t,k}$ are obtained by maximizing the variational lower bound (16) at the end of each E-step, which is guaranteed that the likelihood will not decrease. However, it is difficult to evaluate $q_{t,k}$ by setting the derivative of (16) with respect to the parameters $q_{t,k}$ to zero. Inspired by the setting methods for the covariance matrix of adaptive autoregressive model in [17], we set $q_{t,k}$ to $UC \times trace(\tilde{P}_{t-1,k})/d$, where $trace()$ denotes the trace of matrix, UC as the update coefficients with $0 < UC < 1$. Small UC means the state variables change slowly with time and only a small non-stationary takes place.

3.2 Summary

Here, we give a step-by-step overview of the proposed recursive filtering method:

Initialization: Given a set of measurements $z' = \{Z_t, t = 1, \ldots, L\}$ and feature vectors $h' = \{h_t, t = 1, \ldots L\}$, initialize the parameters $\tilde{m}_{0,k}$ using logistic regression or to random values, and set the parameters $\tilde{P}_{0,k}$ to $q_{0,k}I$, where $q_{0,k}$ are set by hand.

Iterate the VB Approximate Algorithm: Given the approximate posterior distribution $Q(w_{t-1,k})$, first calculate $p(w_{t,k}|Z_{1:t-1}, h_{1:t-1})$ using (10), for $k = 1, \ldots, K$. Then iterate the following a few steps:

Step 1 Calculate $Q(y_{t,k})$ using (10);

Step 2 Calculate $Q(w_{t,k})$ using (9);

Step 3 Calculate $F(Q(\mathrm{w}_t, \mathrm{y}_t), \theta_t)$ using (16) to monitor the convergence of the above iterating procedure.

We close the summary by analyzing the time and space complexity of the method.

Given S samples for the required importance sampling method, the most expensive operation in Step 1 is to compute (15), which takes $O(K(K-1)Sd)$ time. The complexity of Step 2 is $O(Kd^3)$ for computing matrix multiplications in (12). Step 3 has to compute the matrix inversion, hence, takes $O(Kd^3)$.The space complexity of the recursive filtering method is $O(max(S, Kd^2))$.

4 Variational Predictive Distributions of New Inputs

For a new feature vector h_{t+1} at the time step $t+1$, one-step-ahead predictive distribution can be obtained by firstly marginalizing the state variables as follows:

$$\begin{aligned} & p(\mathrm{y}_{t+1}|h_{t+1}, \mathrm{w}_{1:t}) = \int p(\mathrm{y}_{t+1}|h_{t+1}, \mathrm{w}_{t+1}) p(\mathrm{w}_{t+1}|\mathrm{m}^-_{t+1}, \mathrm{P}^-_{t+1}) d\mathrm{w}_{t+1} \\ & = \prod_{k=1}^{K} \int N(y_{t+1,k}|h^T_{t+1} w_{t+1,k}, 1) N(w_{t+1,k}|m^-_{t+1,k}, P^-_{t+1,k}) dw_{t+1,k} \\ & = \prod_{k=1}^{K} N(y_{t+1,k}|h^T_{t+1} m^-_{t+1,k}, v^2_{t+1,k}) \end{aligned} \tag{17}$$

where $v_{t+1,k} = \sqrt{1 + h_{t+1}^T P_{t+1,k}^- h_{t+1}}$. As shown in [15], using the definition:

$$\delta_{t+1,k} \equiv p(Z_{t+1} = k|\mathrm{y}_{t+1}) = \delta(y_{t+1,k} > y_{t+1,i}\ \forall i \neq k)\delta(Z_{t+1} = k)$$

then:

$$\begin{aligned} p(Z_{t+1} = k|h_{t+1}, \mathrm{w}_{1:t}) &= \int \delta_{t+1,k} \prod_{k=1}^{K} N(y_{t+1,k}|h_{t+1}^T m_{t+1,k}^-, v_{t+1,k}) dy_{t+1,k} \\ &= E_{p(u)}\{\prod_{j\neq k} \Phi[uv_{t+1,k} + h_{t+1}^T(m_{t+1,k}^- - m_{t+1,j}^-)/v_{t+1,j}]\} \end{aligned} \tag{18}$$

Constrained by the definition of probabilistic distribution, a proper normalization for the posterior distribution over classes $\{1, \ldots, K\}$ should be adopted:

$$\sum_{k=1}^{K} p(Z_{t+1} = k|h_{t+1}, \mathrm{w}_{1:t}) = 1$$

When labels received are sparse, the classification problem can be reformulated as an incomplete learning problem with missing labels. The one-step-ahead method can also be applied to infer the missing labels. In turn, the predictive labels with the largest posterior probabilities are set to "quasi-targets" for further recursive filtering. The approach can also be viewed as a semi-supervised learning problem of time series. As illustrated in [6], moreover, active label requesting can be adopted when observing labels may be expensive. In practice, successive applying of one-step-ahead for missing labels can bring high cumulative errors by the recursive filtering method proposed as above. Therefore, requesting actively a label is important to the further research.

5 Experiment Results

Despite increasing interest and importance of this topic, there have few publicly available benchmark data sets for dynamic multi-class learning tasks involving non-stationary environments. We present results over three data sets: A synthetic data set,a real data set of electroencephalogram(EEG) and the Waveform data set from the UCI Machine Learning Repository. Some static classifiers, as well as CVFDT and an online linear discriminant classifier (O-LDC) presented in [10], are compared with our proposed approach on the two data sets. CVFDT only reports the one-step-ahead results on the three data sets because it fails to tackle the problems with missing input labels. Although O-LDC is one of the few dynamic multi-class classifiers, it is linear.

5.1 Synthetic Problem

For demonstration purposes, we firstly consider a synthetic four-class problem, where four Gaussian distributions rotate in a circular fashion around a central point $[0, 0]$. Initialization mean points of the four Gaussian distributions locate

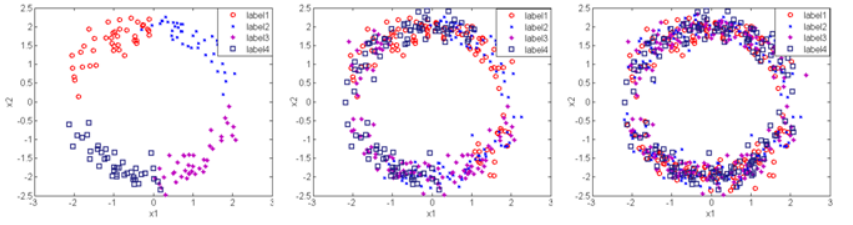


Fig. 2. Four Gaussian distributions rotating in a circular fashion at time step: (a) $t = 160$;(b) $t = 400$; (c) $t = 640$

at the left, upper, right and lower points of a circle whose radian is 2 respectively. Covariance matrices of the Gaussian distributions are equally set to $0.04I_2$. Target classes change with time in an interleaving manner, i.e. $\{1, 2, 3, 4, 1, 2, \cdots\}$. Figure 2 gives three snapshots of the moving-circle class configurations.

The input feature vector h_t at time t in the experiments is represented as $h_t = [x_t^T; 1]^T$ and x_t is a two-dimensional input data. The initialization values of parameters are given as follows: $\tilde{m}_{0,k} \sim N(0, 1)$;$UC = 0.55$.

Results: (1) We performed experiments using first 20 labeled inputs. The remaining 620 inputs were used for testing. One-step-ahead predicted labels were computed by the recursive filtering, the test error was only 1.61%. As a comparison, CVFDT and O-LDA gave the test error rate of 55.8% and 70.8% respectively. We attribute the improvements of performance to the fact that our model takes advantages of non-linear classifier.

(2) Another experiment was performed for missing input labels. Besides the first 40 inputs which must be labeled, 10% to 90% of the remaining 600 inputs were labeled randomly. The predictive results by our proposed method (SDMM) were compared with other three classifiers. Figure 3 illustrated the performances of each classifiers evaluated over 10 independent runs. The parameters of variational Bayesian multinomial probit regression with Gaussian process priors (VBGP) are set to be default [15]. In support vector machines (SVM) with Gaussian kernels, gamma coefficient was set to be 3 according to the best predictive results by observing various numerical experiments under different gamma values ($\gamma = 0.1, 0.5, 1, 2, 3, 5$). We combined the binary SVM classifiers by using 1-vs.-1 scheme [18]. In k-nearest neighbors classifiers, k was set to be 7 according to the best predictive results among $k = 1, 3, 5, 7, 9$.

The left panel of Fig.3 shows that our method (SDMM) takes advantages of sequential dynamic learning. The error rate clearly shows a slow trend to the probabilities of missing labels. Even there are 90% unlabeled test data, the average error rate is only 13%. The other three non-sequential classifiers fail to discrimination. Their error rates have little improvements than that of random labeled procedure.

The results provide the similar conclusion, presented in [6] for the sequential dynamic learning of two-class problems, that the classification performances of the model do not worsen in proportion to the number of missing labels. Although the complexity of the SDMM and corresponding inference algorithms increases, the classification performances with sparse observed labels remain almost unchangeable. Therefore, the results indicate the extension of two-class adaptive classifiers to multi-class environment is feasible.

5.2 Four-Class Motor Imagery EEG Data for the BCI-Competition 2005

BCI aims to establishing a direct connection between the human and the computer, which enables the person to control the computer with the imagination or other mental tasks. Extensive training is required for the adaptation of the user to the designed BCI system. Non-stationary of brain signals during motor imagery poses a challengeable task for the accuracy of BCI control. For investigating the significant challenge, we tried to apply the sequential dynamic multi-class model for a four-class motor imagery task.

The data we use is available at http://www.bbci.de/competition/iii/, whose detailed description of the acquisition procedure is presented in [19]. A four-class classification task including the motor imagery of the left/right hand, one foot, or tongue was provided for the 2005 BCI competition from Graz. For demonstration purposes, we only considered subject k3b, where 90 trials per condition were recorded. According the experimental results in [20], we adopted following feature extraction procedure. Firstly, two monopolar channels ♯3 and ♯34 were selected to perform single-channel analysis. Data within the interval of to 6.8s after down sampling from 250 to 125 samples per second were remained. Then, three-order AAR procedure was calculated for the difference of two monopolar channels of each trail. Finally, 75-dimensional input data for each trail were obtained to construct the input feature vector $h_t = [x_t^T; 1]^T$. Similarly, first 40 inputs and randomly sam-pled inputs from 10% to 90% of the remaining 320 inputs were labeled. $UC = 0.0055$; Weights of the logistic regression classifier on first 40 inputs were taken as the initialization values of $\tilde{m}_{0,k}$.

The predictive performance of our proposed method (SDMM) was also compared with another four classifiers (VBGP, SVM, k-NN and LR). The reported labeling errors are averages over 10 randomly drawn observable and missing data sets. Figure 4 shows that the comparison of the error rate via five different classifiers on the four-class motor imagery EEG data set. Here, gamma for SVM was set to be 0.01 in order to achieve the best results. PCA (principal component analysis) is used to extract 8 principal component features before k-NN classifiers applying to these features. LR stands for Logistic Regression of multi-class learning.

In the right panel of Fig.3, our proposed method (SDMM) has an accuracy improvement of 9% averagely. It is able to achieve higher classification performances than the other four non-sequential models. We use the paired t-test and the p-values for the nine groups under varying fraction of missing labels.

All p-values of pairs between our SDMM method and other methods are less than 0.01 excluding that of pair between the SDMM and SVM on the seventh group (p=0.02053). According to the quantitative results of statistical analysis, the error rate of SDMM is significantly smaller than that of four static classifiers.

It might be argued that the performance of both classifiers almost does not change when the number of missing labels changes. There are two possible reasons: 1) A simple feature extraction approach was presented in our experiments and would lead lower absolute classification accuracies; 2) In BCI, the labels of subject training are not always accurate [8]. In addition, unlike the other classifiers improving performances by extensive cross validation procedures, our method is prone to online computation style.

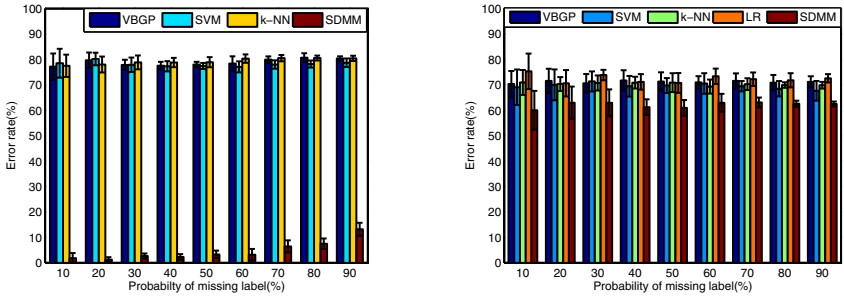


Fig. 3. Percent error rates of VBGP, SVM, k-NN and SDMM corresponding to all of the 9 tests under varying fraction of missing labels. Error bars are standard errors of the mean (SEM) over the ten runs with random realizations of the labeled inputs. The left and right panels show the performance comparision results with different probabilities of missing labels on the synthetic problem and the four-class motor imagery EEG data respectively.

5.3 Waveform Data Set from the UCI Machine Learning Repository

The original Waveform data set consists of 5000 instances in three classes with 21 attributes. To form a large data stream, the data set is repeatedly stacked according the order of attributes. For every segment partitioned by the selected attribute, all instances in the segment are ascendingly sorted according to the value of attribute [21]. Therefore, the Waveform data set has 105000 instances.

For the large scale stream learning problem, SDMM is compared against the popular multi-class data stream classifier CVFDT. The CVFDT algorithm has some parameters to be set. Typical parameter setting used by the corresponding paper and free software can not achieve the low error rate along time for the data set. In our study, we observed that the setting "-tc 0.25 -sc 0.1 -chunk 20" can give the best results. In our SDMM algorithm, we set $UC = 0.0055$.

In Fig.4 the incremental one-step-ahead error rate is illustrated for the two rival algorithms. It is observed that SDMM outperforms CVFDT when the number of instances is less than 50000. However, accuracy of CVFDT improves steadily as more instances are increased. This is explained by the fact that SDMM, as a

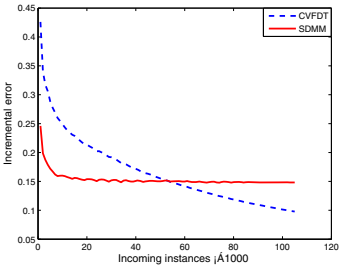


Fig. 4. Incremental one-step-ahead error rate of rival algorithms on Waveform data set

model-based classifier learned by the Bayesian methods, is prone to tackle the small scale learning problems. Another possible reason is that our SDMM algorithm keeps a sliding window with fixed size 1 and reduces the significance of the most recent instances arrived.

6 Conclusions

We have presented the sequential dynamic multi-class model for adaptive learning of data streams. We have adopted the variational Bayesian approach for recursive filtering of the model. Our experiments have shown that this model outperforms the static approach in the presence of time-varying classification decisions and sparse labeled data. In comparison to the well-known multi-class data stream algorithms, we report promising results.

Sequential adaptive learning is important for exploring the nature of non-stationary and non-linear of online learning problems. Future work will focus on developing an advanced filtering method along an adaptive sliding window with varying size. Active label requesting also deserves further study.

Acknowledgments. This study has been supported by the Nation Nature Science Foundation of China 61074113, Shanghai Leading Academic Discipline Project B504, and Fundamental Research Funds for the Central Universities WH0914028.

References

1. Millan, J.D.R.: On the need for on-line learning in brain-computer interfaces. In: Proceedings of the 23rd International Joint Conference on Neural Networks, pp. 2877–2882 (2004)
2. Shenoy, P., Krauledat, M., Blankertz, B., Rao, R.P.N., Muller, K.-R.: Towards adaptive classification for BCI. Journal of Neural Engineering 3, R13–R23 (2006)
3. Hojen-Sorensen P. A.d.F.R, Freitas N., Fog T.: On-line probabilistic classification with particle filters. In: Proceedings of the 2000 IEEE Signal Processing Society Workshop, pp. 386–395 (2000)

4. Penny, W.D., Roberts, S.J.: Dynamic logistic regression. In: International Joint Conference on Neural Network, pp. 1562–1567 (1999)
5. Sykacek, P., Roberts, S.J., Stokes, M.: Adaptive BCI based on variational Bayesian Kalman filtering: an empirical evaluation. IEEE Trans. Biomedical Engineering. 51(5), 719–727 (2004)
6. Lee, S.M., Roberts, S.J.: Sequential dynamic classification using latent variable models. Technical Report PARG 08-02, University of Oxford (2008)
7. Yoon, J.W., Roberts, S.J., Dyson, M., Gan, J.Q.: Sequential Bayesian estimation for adaptive classification. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 601–605 (2008)
8. Yoon, J.W., Roberts, S.J., Dyson, M., Gan, J.Q.: Adaptive classification for brain computer interface systems using sequential Monte Carlo sampling. Neural Networks 22(9), 1286–1294 (2009)
9. Lowne, D.R., Roberts, S.J., Garnett, R.: Sequential non-stationary dynamic classification with sparse feedback. Pattern Recognition 43(3), 897–905 (2010)
10. Kuncheva, L.I., Plumpton, C.O.: Adaptive learning rate for online linear discriminant classifiers. In: da Vitoria Lobo, N., Kasparis, T., Roli, F., Kwok, J.T., Georgiopoulos, M., Anagnostopoulos, G.C., Loog, M. (eds.) S+SSPR 2008. LNCS, vol. 5342, pp. 510–519. Springer, Heidelberg (2008)
11. Tomas, A.: A dynamic Logistic model for combining classifier outputs. DPhil thesis, Lady Margaret Hall, University of Oxford (2008)
12. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proc. ACM SIGKDD 2001, pp. 97–106 (2001)
13. Sarkka, S., Nummenmaa, A.: Recursive noise adaptive Kalman filtering by variational Bayesian approximations. IEEE Trans. Automation Control. 54(3), 596–600 (2009)
14. Albert, J.H., Chib, S.: Bayesian analysis of binary and polychotomous response data. Journal of the American Statistical Association 88(422), 669–679 (1993)
15. Girolami, M., Rogers, S.: Variational Bayesian multinomial probit regression with Gaussian process priors. Neural Computation 18(8), 1790–1817 (2006)
16. Beal, M.J.: Variational algorithms for ap-proximation methods. Ph.D. dissertation, Gatsby Computational Neuroscience Unit, University College London (2003)
17. Schlogl, A.: The electroencephalogram and the adaptive autoregressive model: theory and applications. Shaker Verlag, Aachen (2000)
18. Hsu, C.W., Lin, C.J.: A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 13(2), 415–425 (2002)
19. Blanchard, B., Mueller, K.R., Krusienski, D., Schalk, G., et al.: The BCI competition III: Validating alternative approaches to actual BCI problems. IEEE Trans. Neural Systems and Rehabilitation Engineering. 14(2), 153–159 (2006)
20. Schlogl, A., Lee, F., Bischof, H., Pfurtscheller, G.: Characterization of four-class motor imagery EEG data for the BCI-competition 2005. Journal of Neural Engineering 2, L14–L22 (2005)
21. Hashemi, A., Yang, Y., Mirzamomen, Z., Kangavari, M.: Adapted one-versus-all decision trees for data stream classification. IEEE Transactions on Knowledge and Data Engineering 21(5), 624–637 (2009)

Random Ensemble Decision Trees for Learning Concept-Drifting Data Streams

Peipei Li[1], Xindong Wu[1,2], Qianhui Liang[3], Xuegang Hu[1], and Yuhong Zhang[1]

[1] School of Computer Science and Information Engineering,
Hefei University of Technology, China, 230009
[2] Department of Computer Science, University of Vermont, Vermont, 05405
[3] Hewlett-Packard Labs Singapore

Abstract. Few online classification algorithms based on traditional inductive ensembling focus on handling concept drifting data streams while performing well on noisy data. Motivated by this, an incremental algorithm based on random Ensemble Decision Trees for Concept-drifting data streams (EDTC) is proposed in this paper. Three variants of *random feature selection* are developed to implement split-tests. To better track concept drifts in data streams with noisy data, an improved two-threshold-based drifting detection mechanism is introduced. Extensive studies demonstrate that our algorithm performs very well compared to several known online algorithms based on single models and ensemble models. A conclusion is hence drawn that multiple solutions are provided for learning from concept drifting data streams with noise.

Keywords: Data Stream, Random Decision Tree, Concept Drift, Noise.

1 Introduction

With the development of Web Services technology, streaming data from the Internet span across a wide range of domains, including shopping transactions, Internet search requests, and etc. In contrast to the traditional data sources of data mining, these streaming data present new characteristics as being continuous, high-volume, open-ended and concept drifting. As an important branch of data mining, classification can be solved by inductive learning models[20,23]. Decision trees, due to their advantages, are one of the most popular techniques. They are widely used as a basic model in ensemble classifiers. However, it is still a challenge to learn from data streams with these traditional models. Thus, new decision-tree-based algorithms have been developed for data streams, including a continuously-changing data stream algorithm-CVFDT[16], a Streaming Ensemble Algorithm-SEA[24], a weighted ensemble algorithm[25], a boosting-like method[22] and new variants of Bagging for concept drifting data streams[3]. However, few of them focus on tackling different types of concept drifts while considering the effect from noise in the concept drifting detection.

Contrary to these existing efforts, we introduce a new Ensemble algorithm based on random Decision Trees for Concept drifting data streams with noise

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 313–325, 2011.

(EDTC). Our major contributions are as follows. First, the incremental generation of random decision trees in EDTC is different from all existing random decision trees. Second, we develop three variants of *random feature selection* to determine the split-information. Observations in [10,19] reveal that algorithms with *random feature selection* are more effective and efficient compared to those with informed split-tests in heuristic methods. Lastly, we develop a double-threshold-based mechanism of concept drifting detection in Hoeffding Bounds inequality[13]. Experiments show that EDTC enables tackling concept drifting data streams with certain noisy data compared to other online algorithms.

The rest of this paper is organized as follows. Section 2 reviews the related work. Our EDTC algorithm is described in detail in Section 3. Section 4 provides the experimental study and Section 5 is the summary.

2 Related Work

Many efforts based on ensemble random decision trees have emerged for data stream handling, which are summarized below. An ensembling method [9] based on random decision trees [10] was proposed to improve the accuracy of classification and adapt quickly to sudden drifts. An online algorithm[1] of Streaming Random Forests extended from *Random Forests* was presented for data streams. To further apply this algorithm to tackle concept drifts, an algorithm using an entropy-based drift-detection technique[2] was developed for evolving data streams. In addition, an algorithm of Multiple Semi-Random decision Trees called MSRT[18] was also presented for concept-drifting data streams. It adopts a double-threshold-based mechanism to detect concept drifts.

In contrast to the aforementioned algorithms based on random decision trees, our EDTC algorithm presents three different characteristics. i) Instead of generating all trees with fixed heights in advance[9,10,18], decision nodes are scaled up incrementally after seeing real training data. ii) Various versions of *random feature selection* are explored to solve the split-information in the growing of trees. All variants are different from the *random-feature-selection* mechanism in [2]. iii) A double-threshold-based drift-detection technique is utilized to distinguish concept drifts from noise. This is different from MSRT regarding the evaluation method for error rates of classification and the values of thresholds.

3 The EDTC Algorithm

Our EDTC algorithm aims for concept drifting detection in data streams with noisy data. The processing flow is shown in Algorithm 1. With the continuous incoming of data streams, EDTC will generate N-random decision trees incrementally in various strategies of *random feature selection*, called the component of *GenerateIncrTree*. If the total number of training data arrived amounts to a drifting check period-DP, EDTC will distinguish concept drifts from noisy data streams and adjust the detection mechanism for adaptation to concept drifts.

Algorithm 1: EDTC

Input: Training set: TR=$\{tr_1, tr_2, \cdots, tr_m\}$; Test set: TE=$\{te_1, te_2, \cdots, te_n\}$; Count of attribute dimensions: $|Attr|$: Attribute set: A=$\{a_1, a_2, \cdots, a_{|Attr|}\}$; Maximum height of trees: h_0; Minimum count of split instances: n_{min}; Tree count: N; Memory check period: MP; Drifting check period: DP; Sliding window: SW

Output: Error rate of classification

Procedure EDTC (TR, TE, $|Attr|$, A, h_0, n_{min}, N, MP, DP, SW)

1. for each training instance-$tr_i \in TR$, $1 \leq i \leq m$
2. for each tree-T_k, $k \in \{1, ..., N\}$
3. *GenerateIncTree*(T_k, tr_i, h_0, n_{min}, A, SW);
4. if the count of training instances observed in tree-T_k satisfies the value of DP
5. *DetectDrifts*(DP, SW);
6. if the count of training instances observed in tree-T_k meets the value of MP
7. *AdjustMemory*();
8. return *Voting*(TE);

Function 1: GenerateIncTree{T_k, tr_i, h_0, n_{min}, A, SW}

1. if there is no root in the decision tree T_k
2. Generate the root of T_k;
3. for a training instance tr_i
4. Store the instance tr_i into SW
5. if the number of instances exceeds the size of SW
6. Release the instance in SW by the mechanism of *first in first out*;
7. while the height of tree-$T_k \leq h_0$
8. Sort tr_i into an available node;
9. if the passed current node of $node_{cur}$ is a growing node without splitting
10. Select a split-attribute $A_j \in A$ at $node_{cur}$;
11. if A_j is a numerical attribute
12. Mark the node as *continuous* and update the statistical information;
13. if A_j is a discrete attribute
14. Generate a child by the attribute value of A_j in tr_i;
15. Set this child node to the current node of $node_{cur}$ and go to Step 3;
16. if it is a *continuous* node without cut-point and the count of instances $\geq n_{min}$
17. Do split-test for a split-point;
18. if the node of $node_{cur}$ is a leaf with the height level of h_0
19. Update the information of attribute values relevant to all attributes in A;

This is called the component of *DetectDrifts*. Meanwhile, if the number of training instances arrived is up to the memory check period-MP, it will release the space to avoid space overflow, called *AdjustMemory*. Last, it will adopt a simple voting method to validate the performance itself after training, called *Voting*. In the following subsections, we will describe each component in detail.

Incremental Generation of Decision Trees

In the component of *GenerateIncTree*, to avoid growing sub-branches blindly, each node is generated only if a real training instance arrives. Meanwhile, to obtain the maximum utility of each decision path, the split-test mechanism in the growing of each tree follows that discrete attributes should not be chosen again in a decision path while numerical attributes can be chosen multiple times. In addition, with the continuous incoming data streams, a sliding window is introduced to store training data for future reconstruction of sub-branches. Instances in the window follow the forgetting mechanism of "first in and first out". More details refer to Function 1. In this function, the statistical information involved in Step 12 includes the count of instances, the distribution of class labels and the attribute values relevant to this split-attribute only. Attribute values will be

stored in an ascending order and discritized into min(10, the number of these sequential attribute values) intervals for future split-tests. In Steps 14 and 17, a doubly linked list of *Parent-Children* is created for future bottom-up search, called *PCList*. Each element is a triple, consisting of a self-pointer, a parent pointer and a sibling pointer. It is used in the concept drifting detection.

Random Feature Selection

To solve cut-points for growing nodes with numerical attributes, three *random feature selection* strategies are adopted in this paper. The first one refers to Randomization with the heuristic method in Hoeffding Bounds inequality, called RHB. The second one specifies Randomization with the heuristic method of *Information Gain*, named as RIG, and the third one indicates RANdomization without any heuristic method, called RAN. Details are as follows. In the first strategy, a cut-point is selected from the statistic attribute values of the current split-attribute (e.g., a_j). It is determined in the evaluation function (denoted as $H(\cdot)$) relevant to Information Gain and Hoeffding Bounds inequality. The definition of Hoeffding Bounds inequality is given below. Consider a real-valued random variable r whose range is R. Suppose we have made n independent observations of this variable, and computed their mean $\bar{r}$, which shows that, with probability 1-δ, the true mean of the variable is at least $\bar{r}$-ε,

$$P(r \geq \bar{r} - \varepsilon) = 1-\delta,\ \varepsilon = \sqrt{R^2 ln(1/\delta)/2n} \tag{1}$$

where R is set to log_2(the number of class labels), and n indicates the number of instances required in a split-test (using n_{min} instead). According to Eq.(1), suppose $\Delta H = H(a_j(x)) - H(a_j(y))$ refers to the difference between two candidate cut-points with the highest gain for attribute-a_j. For a given τ[1], if the condition of $\Delta H > \varepsilon$ or $\Delta H \leq \varepsilon < \tau$ holds, the attribute value of the x^{th} cut-point with the highest gain will be selected as the final split-point. In the second strategy, the evaluation function is only related to *Information Gain*. In other words, if satisfying $H(a_j(x)) > H(a_j(y))$, the attribute value of x^{th} cut-point with the highest gain will be selected as the final split-point. However, the third strategy evaluates the cut-point without any heuristic method. That is, it first randomly selects a discretization interval index of attribute values for the current numerical split-attribute, and then specifies the average value of the selected interval as the split-point. In brief, because each strategy implies a certain random characteristic in the feature selection, we call them variants of *random feature selection*[2].

Concept Drifting Detection

A double-threshold-based concept drifting detection mechanism presented in this section is developed to improve the performance in the tracking of concept drifts under noise. Our thought is enlightened from the drifting detection in [11], but a

[1] τ is the threshold that we do not care about which classifier is more accurate, i.e., the case of ties. In general, the value of τ is set to 0.05 [16] [19].

[2] The executable file is available on http://www1.hfut.edu.cn/organ/kddweb/

difference is that thresholds are determined in the Hoeffding Bounds inequality. Details of drifting detection in *DetectDrifts* are as follows. In terms of the structure *PCList*, a bottom-up scanning in each tree starts on decision nodes with the 2^{nd} highest level and ends at the root of the current tree for each drifting check period. More precisely, for each available decision node, we first calculate the sum error rate of classification in Naïve Bayes at its children nodes. Suppose this is the p^{th} check period. Estimation values in the p^{th} and $(p-1)^{th}$ periods are denoted as e_f and e_s respectively. Both variables are considered as independent ones, which follow the Normal distribution. According to the reproducibility of Normal distribution, the linear transformations of these variants (e.g., $\Delta e = e_f - e_s$) also follow this distribution. Considering that the probability distribution is unchanged when the context is static, then the 1- δ confidence interval for Δe is approximately $(\bar{e}_f - \bar{e}_s) \pm c\varepsilon$ according to the inequality of Hoeffding Bounds, where c is a constant, $\bar{e}_f = 1/p \cdot \sum_{k=1}^{p} e_k$, $\bar{e}_s = 1/(p-1) \cdot \sum_{k=1}^{p-1} e_k$, and e_k indicates the error rate in the k^{th} checking period. In our algorithm, we ignore the deviation between $\bar{e}_f$ and $\bar{e}_s$, because with the increasing of the value of p, the difference of $\bar{e}_f - \bar{e}_s$ approaches to zero. In the analysis of the relation among c, Δe and δ, we find that the larger value of c indicates the larger value of Δe and the smaller value of δ while the larger value of Δe indicates the larger deviation in the data distribution during different periods, namely, the more possibility that concept drifts occur (i.e., the larger value of $1 - \delta$). In addition, regarding the noise impact in the concept drifting detection, we use two different values of c (c_1 and c_2, $c_1 < c_2$) to distinguish concept drifts from noise. Hence, we could obtain three cases relevant to concept drifts below.

i) If $\Delta e \leq c_1\varepsilon$, we only consider that a *potential* concept drift is occurring with the confidence 1 - δ_l, where $c_1\varepsilon = 1 \,/\, \exp[\delta_l{}^2 \cdot 2n/R^2]$.

ii) If $\Delta e \geq c_2\varepsilon$, it indicates that a *true* concept drift is involved with the confidence 1 - δ_h, where $c_2\varepsilon = 1 \,/\exp[\delta_h{}^2 \cdot 2n/R^2]$. In this case, sub-trees will be reconstructed using the instances in SW for better tracking concept drifts.

iii) Otherwise, a *plausible* concept drift is considered due to the impact from noise. In the first two cases, n indicates the instance count at the current node.

Handling of Space Overflow

To avoid space overflow in the growing of a decision tree, we provide an approach to space relief in the component of *AdjustMemory*. That is, firstly, stop all undergoing splits of growing nodes and change them into leaves. Secondly, release the storage space of attribute values at decision nodes. It is implemented by traversing from bottom to top using the structure of *PCList*. Lastly, to perform simple pruning, cut off several sub-trees from bottom to top with the roots whose error rates of classification are more than 50%.

Majority Voting for Classification

Regarding the classification in our algorithm, the strategy of majority voting is utilized to classify each test instance. It is implemented after the individual

decision from each tree in the Naïve Bayes method, namely choosing the majority class label as the classification result.

Analysis: Estimation on Generalization Error

In the theorem of generalization errors for a model of random tree ensembling, an upper bound [6] is given in Eq.(2), where $\bar{p}$ indicates the correlation among trees while s indicates the strength of each individual tree.

$$PE \leq \bar{p}(1-s^2)/s^2 = \bar{p}(1/s^2 - 1) \tag{2}$$

This is also suitable for our random ensemble decision trees[15]. According to the definitions on three strategies of *random feature selection* mentioned above, all trees in the ensemble model are created in a completely random selection method on split-attributes, the correlation among trees is hence lower, namely ensuring a smaller value of $\bar{p}$. However, considering the value of s, it is non-deterministic due to the different randomization in each individual model. To evaluate the strength of our model, it is sufficient to take into account the probability that this model is optimal, as the higher predictive accuracy in a model infers the more likelihood that this model is optimal[8]. More specifically, supposing a database has only one relevant attribute a_i and the remaining $|Attr|$-1 attributes are irrelevant, there are two extreme cases as follows.

Case1 (*Best* Case): There are pure discrete-only attributes in the database and all of the discrete attributes are binary. Each decision path from the root to a leaf is completely independent.

Case2 (*Worst* Case): All attributes in the database are numerical. Each node in a tree only generates two children branches at most. The sampling mechanism with replacement is used in the generation of trees.

In both cases, probabilities of an attribute to appear in the L_{th} level of a decision path are defined in Eqs.(3) and (4) respectively (let K=$|Attr|$). Thus,

$$Case1 : p_{max} = \frac{K-1}{K} \cdot \frac{K-2}{K-1} \cdot \cdots \cdot \frac{K-L}{K-L+1} \cdot \frac{1}{K-L} = \frac{1}{|Attr|} \tag{3}$$

$$Case2 : p_{min} = \frac{K-1}{K} \cdot \frac{K-1}{K} \cdot \cdots \cdot \frac{K-1}{K} \cdot \frac{1}{K} = [1 - \frac{1}{|Attr|}]^{L-1} \cdot \frac{1}{|Attr|} \tag{4}$$

we have the least probability for at least one path to involve the only relevant attribute in Eq.(5), namely, our model is optimal with the value of LP at least.

$$LP = 1 - (1 - p_{max})^{N \cdot 2^{h_0} - 1} \tag{5}$$

The generalization errors mentioned above are obtained on the assumption that the distribution of training data is *i.i.d.* However, it is hard to hold in the concept drifting data streams. Thus, regarding the impact from concept drifts, we give an infimum bound of generalization errors for our random ensemble model in Eq. (6), where T_t specifies the current decision tree ensembling generated over the sequential data chunks $\{\theta_k, 1 \leq k \leq t\}$. Each data chunk θ_k contains all instances

in the k^{th} drifting check period. Due to the page limitation, the detailed inference of Eq. (6) is omitted.

$$PE^* \geq P_{(T_{t'},\theta_t)}(LP \leq 0.5) \tag{6}$$

4 Experiments

To validate the efficiency and effectiveness of our EDTC algorithm, several single and ensemble algorithms for concept drifting data streams, including CVFDT[16], HT-DDM (a single Hoeffding Tree with a Drift Detection Method) [11], HT-EDDM (a single Hoeffding Tree with an Early Drift Detection Method) [4], Bag10-ASHT-W+R[3] [3] and MSRT[18], are selected to compare with our algorithm on benchmark concept drifting databases and real streaming data. All experiments are performed on a P4, 3.00GHz PC with 2G main memory, running Windows XP Professional. Algorithms of CVFDT, MSRT and EDTC are developed in C++ while others are coded in Java from the open source MOA[4] [14]. Thus, only those algorithms coded in the same platform with EDTC on the overheads of space and time are contrasted for fair comparisons. In addition, because three strategies of *random feature selection* in EDTC are designed for cut-points of nodes with numerical attributes, only a copy of experimental values is presented for databases with pure discrete-only attributes. All denotations involved in this section are summarized as shown in Table 1.

Parameter Estimation

With respect to the parameters of h_0 and N in EDTC, it is sufficient to ensure better performance for the current ensemble model if satisfying h_0=$|A|/2$ (or 5) and N=10[8,19]. Considering the parameters of n_{min} and δ in the growing of a tree, an experimental conclusion[19] reveals that the model performs sufficiently well if n_{min}=200 and $\delta = 10^{-7}$. Regarding the values of c_1 and c_2, we also use the control limit of 2-σ or 3-σ[11] to partition concept drifts and noise. In EDTC, let δ_l equal to 0.05, namely, the confidence level for drifts is set to 95% while the confidence of 1-δ_h could be more than 99%. In addition, for other parameters in EDTC, they are set to DP=1k[5], MP=500k and SW=10k. However, for the parameters in the comparison algorithms, they follow the default settings involved in [16,11,4,3,18] respectively.

Evaluations on Concept Drifting Data Sets

SEA. It is a well-known data set of concept shift with numerical attributes only[24]. This database consists of a three-dimensional feature space with two

[3] Online Bagging based on ten Adaptive Size Hoeffding Trees, which uses weighted classifiers and replaces oversized trees with new ones.

[4] Massive Online Analysis: a software environment for implementing algorithms and running experiments for online learning from data streams.

[5] The larger the value of DP, the more the frequency of drifting detection. However, this indicates the more probability that false alarms occur and the more time consumption. To trade-off these values, an experimental conclusion is set to DP=1k.

Table 1. Denotations

Symbol	Description
Memory	The memory consumption in a single tree, Unit: (M).
Mean+Variance	The mean error rate of classification+the standard deviation, Unit: (%).
AD	The average number of concept drifting detections.
FA	The probability that false alarms occur in the drifting detection, Unit: (%).
Miss	The number of concepts missed in the drifting detection.
AE	The average count of examples till detection, Unit: (k) (1k=1000).

classes and four concepts. We use the data generator of SEA in MOA to generate a test set with 100k-sized instances and a training set with 400k-sized instances containing four concepts. Both data sets contain 5% class noise.

STAGGER. It is a standard database of concept-shifting data streams to test the abilities of inductive algorithms[21]. This database consists of three attribute values and three alternative underlying concepts. We also use the database generator of STAGGER in MOA to generate a training set with instances of 1000k and a test set with 500k-sized instances respectively. The training set includes 0.1k concepts and each concept contains 10k-sized instances.

KDDCup99. It is a database for network intrusion detection[17], containing 24-class labels and 41-attribute dimensions in total with 34 dimensions of numerical attributes. We select this database because it has been simulated as streaming data with sampling change [27]. The times of concept changes is 36.

Predictive accuracy

In this subsection, two aspects are concerned to evaluate the predictive ability of EDTC. In one dimension, Figures 1-3[6] present the cases of tracking concept drifts on benchmark databases. We can see that false alarms are prone to appear at the beginning of learning from training data. For instance, all false alarms appear before the first concept shift in SEA. 64.3% false alarms appear when learning from only 1/10 training data of STAGGER. This is because there are larger fluctuations of error rates classified in the model learning from insufficient training data. Concept drifts are hence probably considered falsely. Furthermore, Table 2 reports the statistical results of drifting detection, which show that our algorithm with *random feature selection* could detect most of the concept changes in a few instances after a drift occurs. Meanwhile, regarding values of estimation metrics[12], i.e., *FA*, *Miss* and *AE*, EDTC-RAN performs best. In the other dimension, Figure 4 presents predictive results on the test data. The observations are as follows. i) On SEA, EDTC-RHB performs worse than EDTC-RIG and EDTC-RAN on the predictive accuracy by 2%. The best

[6] In Figures 1-3, values of error rate are reported incrementally every 1k-sized training data. Detection positions and false alarms are plotted in symbols of "+" and "∘" respectively while drifting concepts are drawn in dotted lines. In Figure 3, concept drifts indicate the distributions of class labels, which are marked in the right coordinate axis. Figure 2 reports ten concepts of STAGGER only for clear presentation.

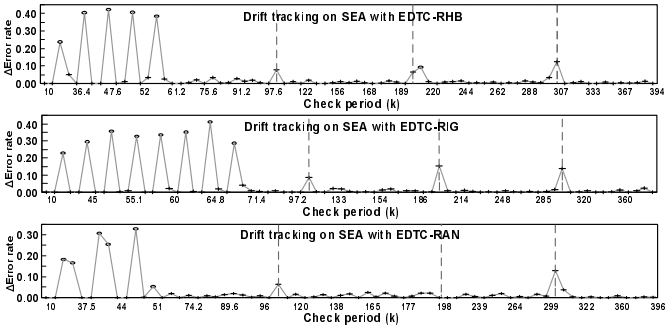


Fig. 1. Drift tracking on the database of SEA

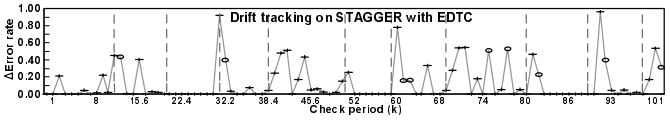


Fig. 2. Drift tracking on the database of STAGGER

Table 2. Drifting detection statistics on training data with concept drifts

	SEA			**STAGGER**	**KDDCup99**		
	EDTC-RHB	EDTC-RIG	EDTC-RAN	EDTC	EDTC-RHB	EDTC-RIG	EDTC-RAN
AD	77	75	69	1019	123	123	126
FA(%)	6.49	10.67	7.25	7.36	7.32	4.87	3.17
Miss	0	0	1	14	8	8	9
AE(k)	7.000	6.930	6.700	1.877	4.166	4.968	1.963

predictive accuracy in EDTC-RIG is lower than CVFDT by 3% while the deviation from Bag10-ASHT-W+R, HT-DDM and HT-EDDM is limited to 1%. ii) On STAGGER, EDTC is superior to all other algorithms. The predictive accuracy is improved by 10% at least. iii) On KDDCup99, predictive accuracies in EDTC-RIG and EDTC-RAN are only lower than that of Bag10-ASHT-W+R by around 4% while they are improved by the range of [14%, 45%] compared to the remaining algorithms. However, it seems abnormal for EDTC-RHB with a more than 80% error rate. This is resulted from the fact that KDDCup99 presents a heavily skewed distribution of class labels. Meanwhile, the constraint of Hoeffding Bounds inequality used in EDTC-RHB impedes the growing of trees. It is hence prone to generate tree stumps, which leads to a poor performance on the predictive ability.

Speed and Space

A set of experiments is conducted to evaluate the overheads of runtime and space in EDTC. Experimental results shown in Table 3 present that our algorithm demands in a light weight way compared to other algorithms. More precisely, on SEA, the lowest time consumption is only 1/2 of that in MSRT and 1/16 of that in CVFDT respectively while the space overhead is no more than 1/38 of

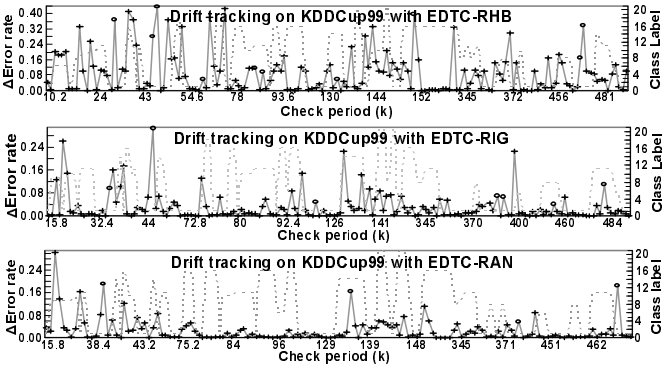


Fig. 3. Drift tracking on the database of KDDCup99

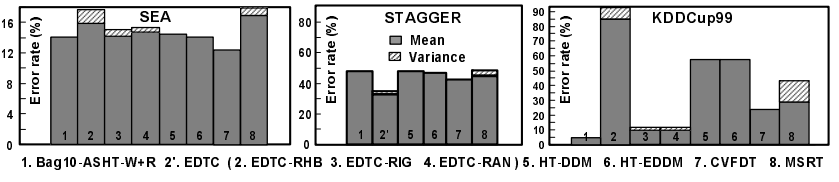


Fig. 4. Classification on databases of STAGGER and KDDCup99

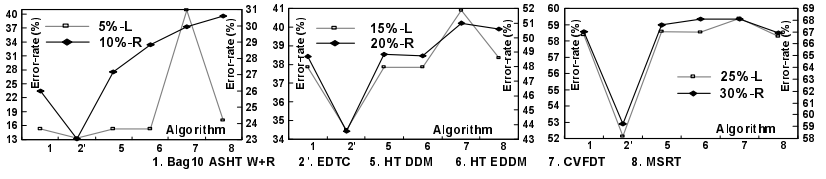


Fig. 5. Classification on databases of LED-noise with drifting attributes

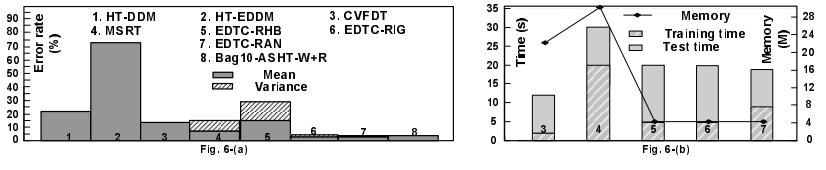


Fig. 6. Classification on the database of Yahoo shopping data

those in CVFDT and MSRT. On STAGGER, EDTC performs as approximately as CVFDT, especially on the time consumption while it outperforms MSRT explicitly on the overheads of runtime and space. However, on KDDCup99, the heaviest overhead of time in EDTC is twice more than that in CVFDT while the space consumption is less than 1/7 of that in CVFDT.

Table 3. Overheads of space and runtime on databases of concept drift

Database	Algorithms	CVFDT	MSRT	EDT-RHB	EDT-RIG	EDT-RAN
SEA	Memory(M)	76	77	<2	<2	<2
	Training+Test time(s)	321+1	38+2	19+1	19+1	19+1
STAGGER	Memory(M)	5	60	<40		
	Training+Test time(s)	21+4	77+13	25+6		
KDDCup99	Memory(M)	29	67	<4	<4	<4
	Training+Test time(s)	75+18	92+504	85+51	75+201	90+118

Robustness

In another dimension, we adopt the noisy database of LED (with 24 dimensions of attributes containing 17 irrelevant attributes and 7 drifting attributes) generated by MOA[14] to validate the robustness to noise in EDTC. This data set contains 1000k-sized training data and 500k-sized test data. Experimental results shown in Figure 5[7] present that EDTC is superior to other algorithms in the resilience to noise. As the noise rates vary from 5% to 30%, the predictive accuracy in EDTC is improved from 2% to 7%. The reason is that EDTC makes use of a semi-random strategy or completely random strategy to select split-features. The process of selecting attributes is independent of the distribution of classes, which reduces the impact of noisy data on classification largely.

Application on Web Shopping Data

In this last subsection, a real data stream from Yahoo shopping databases is obtained via Yahoo web services[26] to verify the feasibility in EDTC. The data set contains 113k-sized records and 22-dimension attributes with 16-dimension numerical ones. To mine the relation between the credibility of merchants and possible factors, we define the attribute of "*OverallRating*" as the class label and divide its values into five class labels. In our experiments, we randomly select 2/3 of total records as the training set and the remaining 1/3 as the test set corresponding to the original distribution of class labels. Experimental results shown in Figure 6 reveal that EDTC-RIG and EDTC-RAN perform as well as Bag10-ASHT-W+R on the predictive accuracy while all of them outperform other algorithms very much. For instance, the lowest predictive accuracy and the highest one in EDTC-RIG are improved by 1.28% and 70.26% respectively. Meanwhile, the maximum time consumption in a single tree is no more than 20s and the space overhead is only 4M at most. These data confirm that EDTC is suitable for handling real streaming data.

5 Conclusion

In this paper, we have proposed an algorithm of random Ensemble Decision Trees for Concept drifting data streams called EDTC. Unlike other random decision trees, three variants of *random feature selection* are developed to determine the cut-points in the growing of decision trees. Meanwhile, two thresholds defined in Hoeffding Bounds inequality are adopted to track concept drifts from noisy

[7] In this figure, the symbol of "L"/"R" refers to the left/right coordinate axis.

data streams. Experimental evaluations show that EDTC performs better than several state-of-the-art online algorithms. An application of EDTC on a real-world Yahoo shopping data has also shown promising results. However, how to adapt to gradual concept drifts is an issue for our future work.

Acknowledgements

This research is supported by the 973 Program of China under award 2009CB326-203, the National Natural Science Foundation of China (NSFC) under grants 60828005, 60975034 and 61005007, the Natural Science Foundation of Anhui Province under grant 090412044 and Special Funds of Thousand Talents Program under grant 2010HGXJ0715.

References

1. Abdulsalam, H., Skillicorn, D.B., Martin, P.: Streaming Random Forests. In: DEAS 2007, pp. 225–232 (2007)
2. Abdulsalam, H., Skillicorn, D.B., Martin, P.: Classifying Evolving Data Streams Using Dynamic Streaming Random Forests. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) DEXA 2008. LNCS, vol. 5181, pp. 643–651. Springer, Heidelberg (2008)
3. Albert, B., Holmes, G., Pfahringer, B., Kirkby, R., Gavald, R.: New Ensemble Methods For Evolving Data Streams. In: KDD 2009, pp. 139–148 (2009)
4. Baena-García, M., Campo-Ávila, J.D., Fidalgo, R., Bifet, A., Gavaldà, R., Morales-Bueno, R.: Early Drift Detection Method. In: ECML PKDD Workshop 2006, pp. 77–86 (2006)
5. Breiman, L.: Bagging Predictors. Machine Learning 24(2), 123–140 (1996)
6. Breiman, L.: Random Forests. Machine Learning 45(1), 5–32 (2001)
7. Dietterich, T.G.: An Experimental Comparison of Three Methods for Constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning 40(2), 139–157 (2000)
8. Fan, W.: On the Optimality of Probability Estimation by Random Decision Trees. In: AAAI 2004, pp. 336–341 (2004)
9. Fan, W.: StreamMiner: A Classifier Ensemble-based Engine to Mine Concept-drifting Data Streams. In: VLDB 2004, pp. 1257–1260 (2004)
10. Fan, W., Wang, H.X., Yu, P.S., Ma, S.: Is Random Model Better? On Its Accuracy and Efficiency. In: ICDM 2003, pp. 51–58 (2003)
11. Gama, J., Medas, P., Castillo, G., Rodrigues, P.P.: Learning with Drift Detection. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 286–295. Springer, Heidelberg (2004)
12. Gama, J., Sebastião, R., Rodrigues, P.P.: Issues in Evaluation of Stream Learning Algorithms. In: KDD 2009, pp. 329–338 (2009)
13. Hoeffding, W.: Probability Inequalities for Sums of Bounded Random Variabless. Journal of the American Statistical Association 58(301), 13–30 (1963)
14. Holmes, G., Kirkby, R., Pfahringer, B.: MOA: Massive Online Analysis (2007), http://sourceforge.net/projects/moa-datastream
15. Ho, T.K.: The Random Subspace Method for Constructing Decision Forests. Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)

16. Hulten, G., Spencer, L., Domingos, P.: Mining Time-Changing Data Streams. In: KDD 2001, pp. 97–106 (2001)
17. KDDCUP99 data set (1999), http://kdd.ics.uci.edu/databases/kddcup99
18. Li, P.P., Hu, X.G., Wu, X.D.: Mining Concept-drifting Data Streams with Multiple Semi-random Decision Trees. In: Tang, C., Ling, C.X., Zhou, X., Cercone, N.J., Li, X. (eds.) ADMA 2008. LNCS (LNAI), vol. 5139, pp. 733–740. Springer, Heidelberg (2008)
19. Li, P., Liang, Q., Wu, X., Hu, X.: Parameter Estimation in Semi-Random Decision Tree Ensembling on Streaming Data. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 376–388. Springer, Heidelberg (2009)
20. Quinlan, R.J.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
21. Schlimmer Jr., J.C., Granger, R.H.: Incremental Learning from Noisy Data. Machine Learning 1(3), 317–354 (1986)
22. Scholz, M., Klinkenberg, R.: Boosting Classifiers for Drifting Concepts. Intelligent Data Analysis (IDA) 11(1), 3–28 (2007)
23. Shafer, J., Agrawal, R., Mehta, M.: SPRINT: A Scalable Parallel Classifier for Data Mining. In: VLDB 1996, pp. 544–555 (1996)
24. Street, W.N., Kim, Y.S.: A Streaming Ensemble Algorithm (SEA) for Large-Scale Classification. In: KDD 2001, pp. 377–382 (2001)
25. Wang, H.X., Fan, W., Yu, P.S., Han, J.W.: Mining Concept-drifting Data Streams Using Ensemble Classifiers. In: KDD 2003, pp. 226–235 (2003)
26. Yahoo! Shopping Web Services, http://developer.yahoo.com/everything.html
27. Yang, Y., Wu, X., Zhu, X.: Combining Proactive and Reactive Predictions for Data Streams. In: KDD 2005, pp. 710–715 (2005)

Collaborative Data Cleaning for Sentiment Classification with Noisy Training Corpus

Xiaojun Wan

Institute of Computer Science and Technology
The MOE Key Laboratory of Computational Linguistics
Peking University, Beijing 100871, China
wanxiaojun@icst.pku.edu.cn

Abstract. Labeled review corpus is considered as a very valuable resource for the task of sentiment classification of product reviews. Fortunately, there are a large amount of product reviews on the Web, and each review is associated with a tag assigned by users to indicate its polarity orientation. We can download such reviews with tags and use them as training corpus for sentiment classification. However, users may assign the polarity tag arbitrarily and inaccurately, and some tags are not appropriate, which results in that the automatically constructed corpus contains many noises and the noisy instances will deteriorate the classification performance. In this paper, we propose the co-cleaning and tri-cleaning algorithms to collaboratively clean the corpus and thus improve the sentiment classification performance. The proposed algorithms use multiple classifiers to iteratively select and remove the most confidently noisy instances from the corpus. Experimental results verify the effectiveness of our proposed algorithms, and the tri-cleaning algorithm is most effective and promising.

1 Introduction

Sentiment classification is a task of identifying the polarity orientation for a given text or document. Usually, the polarity orientation can be either positive or negative. Product review is one of the most popular text genres in previous researches. Various methods have been proposed for addressing this task, including lexicon-based methods and corpus-based methods. In this study, we focus on corpus-based methods for review sentiment classification.

For corpus-based methods, the labeled training corpus is the most valuable resource and the task is usually addressed by training a classifier on the corpus and then applying the classifier to predict the polarity orientation of unlabeled reviews. Generally speaking, there are the following two ways for constructing the training corpus: 1) The corpus is manually labeled by human subjects; 2) The corpus is automatically constructed by using web mining and statistical learning techniques. The first way is very time-consuming and labor-intensive, and the second way is more preferable than the first way. Fortunately, on a few online shopping web sites, users usually write reviews for a particular product

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 326–337, 2011.

and they also attach a tag to each review for indicating the polarity orientation of the review. The reviews and the associated tags can be automatically crawled and used to construct the training corpus. For example, on a popular Chinese online shopping web site (www.amazon.cn), a user can write a review about a particular product, and then assign one or more stars to the product. The number of the stars indicates the review's polarity orientation. Based on our observation and analysis, if the star number is equal to or less than two, the review is negative, and if the star number is equal to or larger than three, the review is positive.

However, some users assign the tags to the reviews in an arbitrary way and not all users assign the tags consistently, and thus a few tags are incorrectly assigned. For example, we manually check the consistency between the polarity orientation of 1000 reviews and the associated stars on the web site (www.amazon.cn), and we find that the stars for 95 reviews are incorrectly assigned, and some negative reviews are assigned with three or more stars, while some positive reviews are assigned with two or less stars. For example, a negative review for Nokia 1616 is "The order made on Nov. 4, 2010 has not shipped yet, and the service by Joyo is lacking"[1], but it is inconsistently assigned with four stars. These reviews are called noisy reviews in the corpus, and they can deteriorate the classifier's classification performance because the proportion of noisy reviews is not small, and current classifiers can not tolerate such a noisy training corpus.

In order to address the above problem, we propose novel algorithms (i.e. co-cleaning and tri-cleaning) to find and remove the noisy reviews from the training corpus, and thus improve the sentiment classification performance based on the cleaned corpus. The proposed algorithms use multiple classifiers to iteratively identify and remove the most confidently noisy reviews. In particular, at each iteration in the co-cleaning and tri-cleaning algorithms, the noisy reviews are identified in a collaborative way by using two or three classifiers to help each other. Experimental results verify the effectiveness of the proposed algorithms. The tri-cleaning algorithm is validated to be the best one for corpus cleaning. The cleaned corpus can be provided as a valuable resource for the community of sentiment analysis.

The rest of this paper is organized as follows: Section 2 introduces related work. Sections 3 and 4 introduce the problem and propose our novel solutions, respectively. The empirical evaluation is presented in Section 5. Lastly, we conclude this paper in Section 6.

2 Related Work

2.1 Sentiment Classification

The methods for document sentiment classification can be generally categorized into lexicon-based and corpus-based. Lexicon-based methods usually involve

[1] It is a manual translation for the original Chinese review.

deriving a sentiment measure for text based on sentiment lexicons [7, 13, 16, 17, 31]. In this section, we focus on corpus-based methods. Corpus-based methods consider the sentiment analysis task as a classification task and they usually use a labeled sentiment corpus to train a sentiment classifier. Since the seminal work of [26], various classification models and linguistic features have been investigated [23, 27, 28, 34]. Most recently, a structured model has been proposed for jointly classifying the sentiment of text at varying levels of granularity [21]. Domain adaptation for sentiment classifiers has been investigated in [4], focusing on online reviews for different types of products. Andreevskaia and Bergler [2] present a new system consisting of the ensemble of a corpus-based classifier and a lexicon-based classifier with precision-based vote weighting. Li et al. [20] propose a novel approach to learn from lexical prior knowledge in the form of domain-independent sentiment-laden terms, in conjunction with domain-dependent unlabeled data and a few labeled documents. Kim et al. [18] propose an approach to utilizing term weights for sentiment analysis tasks and shows how various term weighting schemes improve the performance of sentiment analysis systems. Dasgupta and Ng [6] propose a semi-supervised approach to sentiment classification where they first mine the unambiguous reviews using spectral techniques and then exploit them to classify the ambiguous reviews via a novel combination of active learning, transductive learning, and ensemble learning. Chinese sentiment analysis has also been studied [19, 30] and most such work uses similar lexicon-based or corpus-based methods for Chinese sentiment classification. In the most recent years, several studies have been performed to leverage rich English resources for sentiment analysis in other languages [3, 22, 32, 33].

2.2 Data Cleaning

Data cleaning is a traditional research area in the fields of database and data mining [12]. Several works have used data cleaning techniques in the filed of computational linguistics. For example, previous works have used task-dependent or task-independent data cleaning techniques for the NLP tasks of POS tagging [1, 8, 9, 25], verb modality identification [24], PP-attachment [1] and word segmentation [29]. Data cleaning methods have also been used for improving text categorization. Fukumoto and Suzuki [11] address the problem of dealing with category annotation errors which deteriorate the overall performance of text classification, by integrating information from two different classification algorithms: NB and SVM. The proposed method is strictly learner-dependent, because it only works with SVMs as learners, and it is limited to cleaning the support vectors. Esuli and Sebastiani [10] present different techniques for performing training data cleaning in the context of boosting-based learning methods. The techniques correspond to different ways of estimating the likelihood of being mislabeled for each training instance. They evaluate them on two widely used text categorization benchmarks by their capability of spotting misclassified texts purposefully inserted in the training set. And the confidence-based technique has the overall best performance.

3 Problem Statement

Given an crawled sentiment corpus consisting of N reviews and the associated polarity tags, we denote the corpus as $C = \{< x_i, y_i > | i = 1, 2, ..., N\}$, where x_i is a review in the review set X, y_i is the user-assigned polarity tag. We let $y_i = +1$ iff x_i is a positive review and $y_i = -1$ iff x_i is a negative review. Based on the training corpus C, a sentiment classifier $f_C : X \rightarrow R$ can be learned by using a basic classifier f, and the classifier's prediction value for an unlabeled review x_i is denoted as $f_C(x_i)$. The polarity orientation of x_i is finally obtained based on $f_C(x_i)$. In this study, we use the SVM classifier as f, and thus the sign of $f_C(x_i)$ indicates the polarity orientation of x_i, i.e., x_i is positive iff $sign(f_C(x_i))$ is +1, and x_i is negative iff $sign(f_C(x_i))$ is -1. The performance of the classifier on a test set T is measured as $A(f_C, T)$.

Because there are many noisy instances in the original corpus C, the learned classifier is deemed to be not very effective and the performance value $A(f_C, T)$ is deemed to be not high. The problem is how to clean the original corpus C by removing the noisy instances R^{noise} from C. The cleaned corpus is denoted as $C^* = C - R^{noise}$, and we expect that the performance of the classifier learned on C^* can be improved, i.e., $A(f_{C^*}, T) > A(f_C, T)$.

4 The Data Cleaning Algorithms

4.1 Overview

In this study, we propose three algorithms for training data cleaning in the task of sentiment classification. The common aim of the algorithms is to find the most confidently mislabeled instances from the original corpus and then remove them. The first algorithm (Self-Cleaning) is an improvement of the previous confidence-based technique in [10], and it leverages only one classifier for identifying and removing noisy instances in an iterative way. The second algorithm (Co-Cleaning) is inspired by the co-training algorithm [5], and it leverages two classifiers to collaboratively find the noisy instances for each other. The third algorithm (Tri-Cleaning) is an improvement of the second algorithm, and it leverages three classifiers for collaborative data cleaning. The details of the three techniques are described in next sections, respectively.

Note that all the three algorithms require a basic text classifier and they are independent of a specific classifier. That is to say, any existing text classifier can be used in the three algorithms. Typical text classifiers include Support Vector Machine (SVM), Naïve Bayes (NB), Maximum Entropy (ME), K-Nearest Neighbor (KNN), etc. In this study, we adopt the widely-used SVM classifier [15] with the linear kernel and default parameter values. Viewing input data as two sets of vectors in a feature space, SVM constructs a separating hyperplane in the space by maximizing the margin between the two data sets. And the features used for sentiment classification include all unigrams and bigrams, and the feature weight is simply set to term frequency as in [32]. Note that for Chinese text, a unigram refers to a Chinese word and a bigram refers to two adjacent Chinese

words after word segmentation. The sign of the SVM prediction value is used for predicting the polarity labels and the absolute value is used for indicating the confidence level of the prediction.

After we obtain the cleaned corpus by applying one of the above data cleaning algorithms, we can learn a SVM classifier based on the cleaned corpus, and then apply the learned classifier to predict the polarity orientation for the reviews in the test set.

4.2 Self-cleaning

This algorithm is an extension of the previous confidence-based technique in [10] by iteratively applying the confidence-based technique to identify and remove the noisy instances in the corpus. It leverages only one classifier to iteratively train on the current corpus and then select and remove the noisy instances to further refine the corpus.

Given the original corpus $C = \{< x_i, y_i > | i = 1, 2, ..., N\}$, the refined corpus $C^* = C$, and the basic learner f, the self-cleaning algorithm loops for I iterations as follows:

1. Learn the classifier f_{C^*} from C^* using the basic learner f;
2. Use f_{C^*} to label all the reviews in C^*;
3. Identify from C^* the potentially noisy instances whose predicted labels are inconsistent with the original labels: $R = \{< x_i, y_i > | f_{C^*}(x_i)y_i < 0\}$;
4. Rank the instances in R in decreasing order of the confidence value $|f_{C^*}(x_i)|$ and select the top k instances into R^{noise};
5. Remove the instances in R^{noise} from C^*, i.e. $C^* = C^* - R^{noise}$;

In the algorithm, I is a parameter controlling the iteration number, and k is a parameter controlling the number of noisy instances removed at each iteration. We can deduce that at most $I \times k$ instances can be removed from the original corpus.

The rationale of the algorithm lies in that the noisy instances can be selected based on the prediction values of the learner, and after each iteration, the learner can be refined and improved, and its predictions are more reliable, and thus the corpus can be further refined.

Finally, a classifier is learned based on C^* and then the classifier is applied to predict for the reviews in the test set.

4.3 Co-cleaning

In the above self-cleaning algorithm, the classifier is learned based on the noisy corpus and then the classifier is applied to predict for each review in the same corpus. Thus some mislabeled reviews can be potentially predicted to be consistent with their original labels because they have been already used for training the classifier. We call it as "noise fitting".

In order to address the above problem of the self-cleaning algorithm, we propose the co-cleaning algorithm by splitting the original corpus into two sets and

leveraging two classifiers to collaboratively find noisy instances for each other. The proposed co-cleaning algorithm is inspired by the famous co-training algorithm [5]. The co-training algorithm is a typical bootstrapping method, and it starts with a set of labeled data, and increase the amount of annotated data using some amounts of unlabeled data in an incremental way. One important aspect of co-training is that two views are required for co-training to work, and two classifiers are learned based on the two views and they choose confident training examples from the unlabeled data for each other. Different from the co-training algorithm, our proposed co-cleaning algorithm aims to use one classifier's prediction results to help identify and remove noisy instances from the training set for the other classifier, and the co-cleaning algorithm do not require two independent views of the data.

Given the original corpus $C = \{< x_i, y_i > | i = 1, 2, ..., N\}$ and the basic learner f, the co-cleaning algorithm first randomly split C into two sets C_1, C_2 with equal sizes, and we have $C = C_1 \cup C_2$. Two refined training sets are defined as $C_1^* = C_1$, $C_2^* = C_2$. The algorithm then loops for I iterations as follows:

1. Learn the first classifier $f_{C_1^*}$ from C_1^* using the basic learner f;
2. Learn the second classifier $f_{C_2^*}$ from C_2^* using the basic learner f;
3. Refine C_1^* as follows:

a) Use $f_{C_2^*}$ to label all the reviews in C_1^*;

b) Identify from C_1^* the potentially noisy instances whose predicted labels are inconsistent with the original labels: $R_1 = \{< x_i, y_i > | f_{C_2^*}(x_i) y_i < 0 \ and \ < x_i, y_i > \in C_1^*\}$;

c) Rank the instances in R_1 in decreasing order of the confidence value $|f_{C_2^*}(x_i)|$ and select the top p instances into R_1^{noise};

d) Remove the instances in R_1^{noise} from C_1^*, i.e. $C_1^* = C_1^* - R_1^{noise}$;

4. Similarly refine C_2^* by using $f_{C_1^*}$;

In the algorithm, I is a parameter controlling the iteration number, and p is a parameter controlling the number of noisy instances removed from each set at each iteration. And we can deduce that at most $2 \times I \times p$ instances can be removed from the original corpus.

The rationale of the algorithm lies in that the noisy instances in one review set are selected based on the prediction values of the learner trained on the other review set, and after a few iterations, the two learners can be refined and improved, and thus their predictions are more reliable for each other. We believe that the prediction values are more reliable than that in the self-learning algorithm, because the training set for one classifier is different from the review set to be predicted by the classifier in the co-cleaning algorithm.

Finally, a classifier is learned based on $C_1^* \bigcup C_2^*$ and then the classifier is applied to predict for the reviews in the test set.

4.4 Tri-cleaning

In the above co-cleaning algorithm, the prediction of a review in one set is decided only by one another classifier, which may be potentially not very reliable, because the classifier is still trained on a noisy corpus.

In order to address the above problem of the co-cleaning algorithm, we further propose the tri-cleaning algorithm by splitting the original corpus into three sets and training three classifiers, and then using the voting of two classifiers to find noisy instances in the third set. The underlying idea is that the prediction based on two classifiers is usually more reliable than that based on only one classifier.

Given the original corpus $C = \{< x_i, y_i > | i = 1, 2, ..., N\}$ and the basic learner f, the tri-cleaning algorithm first randomly split C into three sets C_1, C_2, C_3 with equal sizes, and we have $C = C_1 \bigcup C_2 \bigcup C_3$. Three refined training sets are defined as $C_1^* = C_1$, $C_2^* = C_2$, $C_3^* = C_3$. The algorithm then loops for I iterations as follows:

1. Learn the first classifier $f_{C_1^*}$ from C_1^* using the basic learner f;
2. Learn the second classifier $f_{C_2^*}$ from C_2^* using the basic learner f;
3. Learn the third classifier $f_{C_3^*}$ from C_3^* using the basic learner f;
4. Refine C_1^* as follows:

a) Use $f_{C_2^*}$ and $f_{C_3^*}$ to label all the reviews in C_1^*, respectively;

b) Identify from C_1^* the potentially noisy instances whose labels are consistently predicted by the two classifiers, but inconsistent with the original labels: $R_1 = \{< x_i, y_i > | f_{C_2^*}(x_i) f_{C_3^*}(x_i) > 0,\ and\ f_{C_2^*}(x_i) y_i < 0, f_{C_3^*}(x_i) y_i < 0,\ and\ < x_i, y_i > \in C_1^*\}$;

c) Rank the instances in R_1 in decreasing order of the confidence value $|f_{C_2^*}(x_i)| + |f_{C_3^*}(x_i)|$, and select the top m instances into R_1^{noise};

d) Remove the instances in R_1^{noise} from C_1^*, i.e. $C_1^* = C_1^* - R_1^{noise}$;

5. Similarly refine C_2^* by using $f_{C_1^*}$ and $f_{C_3^*}$;
6. Similarly refine C_3^* by using $f_{C_1^*}$ and $f_{C_2^*}$;

In the algorithm, I is a parameter controlling the iteration number, and m is a parameter controlling the number of noisy instances removed from each set at each iteration. During each iteration, at most $3 \times m$ noisy instances can be removed, and thus we can deduce that at most $3 \times I \times m$ instances can be removed from the original corpus.

The rationale of the algorithm lies in that the noisy instances in one review set are selected based on the "voting" of two learners trained on the other two review sets. We believe that the predicted labels are more reliable than that in the co-cleaning algorithm, because the labels are voted by two classifiers, and thus in the tri-cleaning algorithm, the noisy instances selected from one set by the other two classifiers can be more trusted.

Finally, a classifier is learned based on $C_1^* \bigcup C_2^* \bigcup C_3^*$ and then the classifier is applied to predict for the reviews in the test set.

5 Empirical Evaluation

5.1 Evaluation Setup

The training review set and the test review set were constructed as follows:

Original Training Set: We crawled a large number of product reviews and their associated tags from a popular Chinese online shopping web site -AmazonChina

(www.amazon.cn). The dataset consists of 45898 positive reviews and 24146 negative reviews. The reviews are about various products, such as consumer electronics, mobile phones, digital products, books, and so on. The polarity tag of each review was automatically judged by the number of the assigned stars of the review. If the star number is equal to or less than two, the review is labeled as negative, and otherwise, the review is labeled as positive.

Test Set: We used the same test set as in [33]. The test set contained 886 product reviews downloaded from another popular Chinese IT product web site IT168 (www.it168.com). The sentiment polarity of each review was manually labeled. The dataset consisted of 451 positive reviews and 435 negative reviews. The reviews focused on IT products.

We used the standard precision, recall and F-measure to measure the performance of positive and negative class, respectively, and used the accuracy metric to measure the overall performance of the system. The metrics are defined the same as in general text categorization.

In the experiments, the proposed algorithms are compared with the following three baseline methods:

Baseline1 (Lexicon-Based): It is a lexicon-based method used in [32]. The semantic orientation value for a review is computed by summing the polarity values of all words in the review, making use of both the word polarity defined in the positive and negative lexicons and the contextual valence shifters defined in the negation and intensifier lexicons. The lexicons are derived from HowNet[2].

Baseline2 (NoCleaning): It directly uses the original training corpus to train the SVM classifier, without training data cleaning.

Baseline3 (BasicCleaning): It use a basic cleaning method to remove potentially noisy instances from the original training set, and then learn a SVM classifier based on the refined training corpus. The basic training data cleaning method is similar to the confidence-based technique in [10]. The method first identifies the potentially mislabeled instances whose $sign(f_C(x_i))$ is not equal to the originally label tag, i.e., $f_C(x_i)y_i < 0$. Then the instances are sorted by decreasing order of $|f_C(x_i)|$. And the top n instances in the ranked list are removed from the original corpus. n is empirically set to 1000 when the performance is the best.

5.2 Evaluation Results

In the experiments, for the three proposed algorithms, we let $k = 2p = 3m$ to guarantee that the total numbers of nosy instances to be removed for the three algorithms are equal after each iteration. We investigate three typical sets of parameter values: $(k = 300, p = 150, m = 100)$, $(k = 150, p = 75, m = 50)$, $(k = 90, p = 45, m = 30)$. We report the best accuracy for each algorithm after a few iterations. Table 1 shows the comparison results. The actual iteration numbers for the algorithms are given in the table.

[2] http://www.keenage.com/html/e_index.html

Table 1. Comparison results

Method	F-measure(Positve)	F-measure(Negative)	Total Accuracy
Self-Cleaning($k = 300; I = 7$)	0.827	0.771	0.802
Co-Cleaning($p = 150; I = 7$)	0.829	0.773	0.805
Tri-Cleaning($m = 100; I = 13$)	0.834	0.790	0.815
Self-Cleaning($k = 150; I = 15$)	0.829	0.775	0.806
Co-Cleaning($p = 75; I = 34$)	0.829	0.784	0.809
Tri-Cleaning($m = 50; I = 25$)	**0.835**	**0.792**	**0.816**
Self-Cleaning($k = 90; I = 25$)	0.829	0.775	0.806
Co-Cleaning($p = 45; I = 55$)	0.832	0.786	0.812
Tri-Cleaning($m = 30; I = 44$)	**0.835**	**0.792**	**0.816**
BasicCleaning	0.821	0.756	0.793
NoCleaning	0.815	0.733	0.781
Lexicon-based	0.786	0.677	0.743

Seen from the table, we can see that all the data cleaning algorithms (including the BasicCleaning algorithm) can improve the classification accuracy. And the proposed three data cleaning algorithms can outperform the BasicCleaning algorithm. In particular, the proposed three algorithms can significantly outperform the baseline NoCleaning method by using sign test, and the tri-cleaning algorithm can significantly outperform the BasicCleaning algorithm by using sign test. For the proposed three algorithms, we can see that the co-cleaning algorithm outperforms the self-cleaning algorithm, and the tri-cleaning algorithm outperforms the co-training algorithm. The results demonstrate that our proposed algorithms are effective and the tri-cleaning algorithm is the best one for training data cleaning because two classifiers' voting is more reliable than one single classifier's prediction when the training corpus contains noises.

We further investigate how the classification performance of the proposed three algorithms is influenced by the iteration number I. Figures 1, 2, 3 present the accuracy curves of the three algorithms with the three parameter settings, respectively. Note that when I is equal to 0, the proposed three algorithms are actually the same with the baseline NoCleaning method.

Seen from the figures, the accuracy values of the three algorithms first increase with the iteration number, and then tend to decrease after a few iterations. The performance curves are very similar to that for the original co-training algorithm. The reason is that when the iteration number is large, the algorithms may incorrectly select and remove some normal instances from the training corpus, after all the noisy instances have been removed.

We can also see that the performance of the self-cleaning algorithm drops more quickly than the other two algorithms, and the co-cleaning algorithm can outperform the self-cleaning algorithm after a few iterations. The tri-cleaning algorithm can almost always outperform the self-cleaning and co-cleaning algorithms. The results further demonstrate that the co-cleaning algorithm is more effective and robust than the self-cleaning algorithm, and the tri-cleaning algorithm is more effective and robust than the self-cleaning and co-cleaning algorithms.

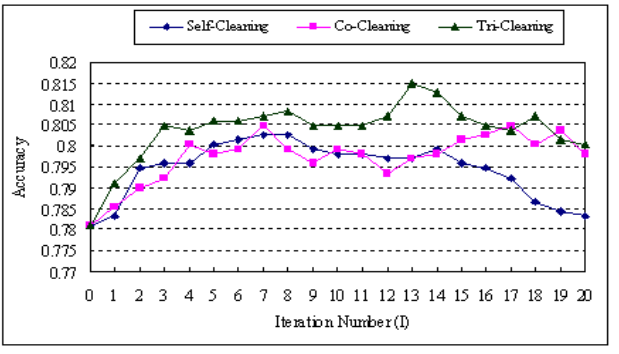


Fig. 1. Accuracy $(k = 300, p = 150, m = 100)$ vs. Iteration number

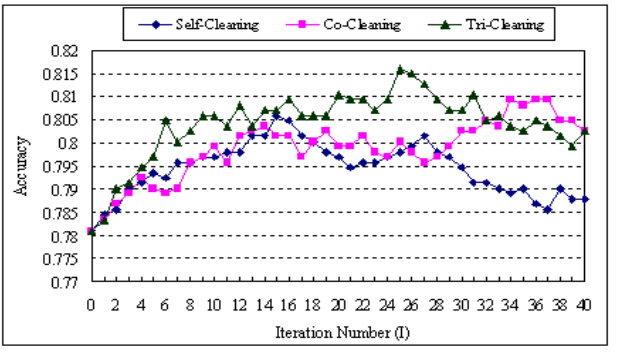


Fig. 2. Accuracy $(k = 150, p = 75, m = 50)$ vs. Iteration number

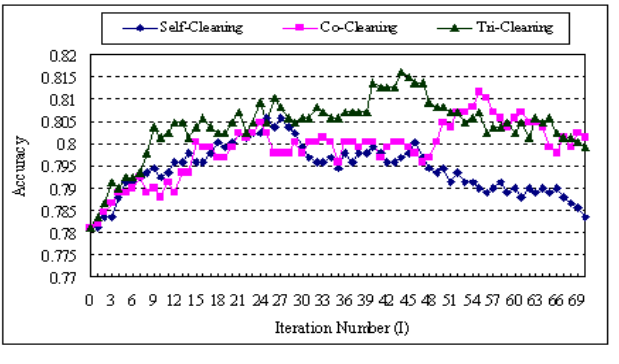


Fig. 3. Accuracy $(k = 90, p = 45, m = 30)$ vs. Iteration number

6 Conclusion and Future Work

In this paper, we propose novel algorithms for finding and removing noisy instances in the training corpus automatically crawled on the web. The three algorithms can improve the sentiment classification accuracy. The proposed tri-cleaning algorithm is the best choice in the experiments.

Though we focus on the task of sentiment classification in this study, the proposed algorithms can also be used for the more general text categorization task, and we will investigate to apply the proposed algorithms for general text categorization tasks to further demonstrate the robustness of the algorithms.

Acknowledgements

This work was supported by NSFC (60873155), Beijing Nova Program (2008B03) and NCET (NCET-08-0006).

References

1. Abney, S., Schapire, R.E., Singer, Y.: Boosting applied to tagging and PP attachment. In: Proceedings of EMNLP/VLC 1999 (1999)
2. Andreevskaia, A., Bergler, S.: When specialists and generalists work together: overcoming domain dependence in sentiment tagging. In: Proceedings of ACL-HLT 2008 (2008)
3. Banea, C., Mihalcea, R., Wiebe, J., Hassan, S.: Multilingual subjectivity analysis using machine translation. In: Proceedings of EMNLP 2008 (2008)
4. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: Proceedings of ACL 2007 (2007)
5. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with cotraining. In: Proceedings of COLT 1998 (1998)
6. Dasgupta, S., Ng, V.: Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification. In: Proceedings of ACL-IJCNLP 2009 (2009)
7. Devitt, A., Ahmad, K.: Sentiment polarity identification in financial news: a cohesion-based approach. In: Proceedings of ACL 2007 (2007)
8. Dickinson, M., Meurers, W.D.: Detecting errors in part-of-speech annotation. In: Proceedings of EACL 2003 (2003)
9. Eskin, E.: Detecting errors within a corpus using anomaly detection. In: Proceedings of NAACL 2000 (2000)
10. Esuli, A., Sebastiani, F.: Training data cleaning for text classification. In: Azzopardi, L., Kazai, G., Robertson, S., Rüger, S., Shokouhi, M., Song, D., Yilmaz, E. (eds.) ICTIR 2009. LNCS, vol. 5766, pp. 29–41. Springer, Heidelberg (2009)
11. Fukumoto, F., Suzuki, Y.: Correcting category errors in text classification. In: Proceedings of COLING 2004 (2004)
12. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2001)
13. Hiroshi, K., Tetsuya, N., Hideo, W.: Deeper sentiment analysis using machine translation technology. In: Proceedings of COLING 2004 (2004)

14. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proceedings of ICML 1999 (1999)
15. Joachims, T.: Learning to classify text using support vector machines. Dissertation. Kluwer, Dordrecht (2002)
16. Kennedy, A., Inkpen, D.: Sentiment classification of movie reviews using contextual valence shifters. Computational Intelligence 22(2), 110–125 (2006)
17. Kim, S.-M., Hovy, E.: Determining the sentiment of opinions. In: Proceedings of COLING 2004 (2004)
18. Kim, J., Li, J.-J., Lee, J.-H.: Discovering the discriminative views: measuring term weights for sentiment analysis. In: Proceedings of ACL-IJCNLP 2009 (2009)
19. Li, J., Sun, M.: Experimental study on sentiment classification of Chinese review using machine learning techniques. In: Proceeding of IEEE-NLPKE 2007 (2007)
20. Li, T., Zhang, Y., Sindhwani, V.: A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In: Proceedings of ACL-IJCNLP 2009 (2009)
21. McDonald, R., Hannan, K., Neylon, T., Wells, M., Reynar, J.: Structured models for fine-to-coarse sentiment analysis. In: Proceedings of ACL 2007 (2007)
22. Mihalcea, R., Banea, C., Wiebe, J.: Learning multilingual subjective language via cross-lingual projections. In: Proceedings of ACL 2007 (2007)
23. Mullen, T., Collier, N.: Sentiment analysis using support vector machines with diverse information sources. In: Proceedings of EMNLP 2004 (2004)
24. Murata, M., Utiyama, M., Uchimoto, K., Isahara, H., Ma, Q.: Correction of errors in a verb modality corpus for machine translation with a machine-learning method. ACM Transactions on Asian Language Information Processing 4(1), 18–37 (2005)
25. Nakagawa, T., Matsumoto, Y.: Detecting errors in corpora using support vector machines. In: Proceedings of COLING 2002 (2002)
26. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: Proceedings of EMNLP 2002 (2002)
27. Pang, B., Lee, L.: A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of ACL 2004 (2004)
28. Read, J.: Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In: Proceedings of ACL 2005 (2005)
29. Shinnou, H.: Detection of errors in training data by using a decision list and Adaboost. In: Proceedings of the IJCAI 2001, Workshop on Text Learning Beyond Supervision (2001)
30. Tsou, B.K.Y., Yuen, R.W.M., Kwong, O.Y., La, T.B.Y., Wong, W.L.: Polarity classification of celebrity coverage in the Chinese press. In: Proceedings of International Conference on Intelligence Analysis (2005)
31. Turney, P.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of ACL 2002 (2002)
32. Wan, X.: Using bilingual knowledge and ensemble techniques for unsupervised Chinese sentiment analysis. In: Proceedings of EMNLP 2008 (2008)
33. Wan, X.: Co-training for cross-lingual sentiment classification. In: Proceedings of ACL-IJCNLP 2009, pp. 235–243 (2009)
34. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In: Proceedings of HLT/EMNLP 2005 (2005)

Using Constraints to Generate and Explore Higher Order Discriminative Patterns*

Michael Steinbach, Haoyu Yu, Gang Fang, and Vipin Kumar

University of Minnesota,
Minneapolis, MN, USA
{steinbach,gangfang,kumar}@cs.umn.edu
haoyu@msi.umn.edu
http://www.cs.umn.edu/~kumar

Abstract. Discriminative pattern mining looks for association patterns that occur more frequently in one class than another and has important applications in many areas including finding biomarkers in biomedical data. However, finding such patterns is challenging because higher order combinations of variables may show high discrimination even when single variables or lower-order combinations show little or no discrimination. Thus, generating such patterns is important for evaluating discriminative pattern mining algorithms and better understanding the nature of discriminative patterns. To that end, we describe how such patterns can be defined using mathematical constraints which are then solved with widely available software that generates solutions for the resulting optimization problem. We present a basic formulation of the problem obtained from a straightforward translation of the desired pattern characteristics into mathematical constraints, and then show how the pattern generation problem can be reformulated in terms of the selection of rows from a truth table. This formulation is more efficient and provides deeper insight into the process of creating higher order patterns. It also makes it easy to define patterns other than just those based on the conjunctive logic used by traditional association and discriminant pattern analysis.

Keywords: association analysis, frequent patterns, discriminative pattern mining, linear programming, synthetic data generation.

1 Introduction

Given a transaction data set where the transactions have class labels, discriminative pattern mining looks for association patterns that occur more frequently in one of the classes than the others. Although such patterns may not cover all the transactions in a data set, they can provide useful classification rules if the discrimination provided by the patterns is accurate enough for the application under consideration. One potentially useful area for discriminative patterns is

* This work was supported by NSF grant IIS-0916439. Computing resources were provided by the Minnesota Supercomputing Institute.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 338–350, 2011.

Table 1. Values of three binary variables and their combinations. The first 10 rows belong to one class (cases), while the last 10 belong to another (controls). The last row is the number of ones in cases minus the number of ones in controls.

	x_1	x_2	x_3	x_1x_2	x_1x_3	x_2x_3	$x_1x_2x_3$
Cases	1	1	1	1	1	1	1
	1	0	0	0	0	0	0
	1	1	1	1	1	1	1
	0	1	0	0	0	0	0
	0	1	0	0	0	0	0
	0	0	1	0	0	0	0
	0	0	1	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	1	0	0	0	0	0	0
Controls	1	0	1	0	1	0	0
	0	1	1	0	0	1	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	1	1	0	1	0	0	0
	1	0	1	0	1	0	0
	0	1	1	0	0	1	0
	1	1	0	1	0	0	0
	0	0	0	0	0	0	0
ones in cases - ones in controls	0	0	0	0	0	0	2

medical studies that involve a set of subjects that are divided into cases (those with a specific disease) and controls (those without the disease). In such situations, any discriminative patterns that are discovered can help identify important factors in the disease and/or its development. The discovery of such *biomarkers* is often of great benefit.

However, finding such patterns is challenging because higher order combinations of individual variables may show high discrimination even when single variables or lower-order combinations only show little or no discrimination. To illustrate, Table 1 shows the values of three binary variables (x_1, x_2, x_3) for 10 cases and controls, along with their combinations: x_1x_2, x_1x_3, x_2x_3, and $x_1x_2x_3$, which are the logical *and* of the indicated variables. The last row in the table shows the difference in the occurrence of ones between cases and controls. All single variables, and the pairs they form, show no discrimination between cases and controls, but the combination of all three variables does. If ones indicate the presence of a particular genetic characteristic, then this may indicate, for example, that the presence of all three genetic factors is necessary for the development of a disease.

It is easy to construct examples of such patterns involving only two variables, e.g., let both variables have 5 ones in both cases and controls, but let the ones in the two variables overlap completely in cases and not at all in controls. However, it becomes more challenging for patterns of size 3 and higher. There may even be some question about the existence of such patterns. Examples of higher order patterns that have better discrimination than any lower order pattern composed of the same variables can indeed be found for sets of variables beyond 3, although as later discussion will show, the difference between the discriminative power of a combination of binary variables and its subpatterns decreases as the number

of variables increases.[1] To address such fundamental questions concerning the nature of discriminative patterns and provide patterns for testing the performance of current discriminative pattern finding algorithms, this paper proposes a constraint-based approach to specifying and generating such higher order patterns. In the rest of the paper, *higher order pattern* will refer *only* to those higher order patterns having more discriminative power than any of their subsets.

Although there are synthetic data generators for ordinary association patterns, such as the IBM Quest Market-Basket Synthetic Data Generator described in [1], we know of no such generator for the higher order discriminative patterns we have been discussing. However, during the course of our research into algorithms on discriminative pattern mining, we encountered the need for such a capability. As a result, we developed the constraint based approach described in this paper. More specifically, we realized that such patterns can be defined using mathematical constraints which are then solved with widely available software that uses linear programming techniques (or extensions of it) to find solutions to resulting optimization problems. Despite the challenges in creating the proper models and limitations on the size of patterns it is feasible to produce, the desired example patterns can usually be generated (often in a relatively short time, i.e., usually seconds or minutes) for patterns up to size 9 or 10.[2]

The insights that can be gained into the nature of higher order patterns are, however, perhaps even more important than generating higher order discriminative patterns for testing discriminative pattern finding algorithms. For instance, through experimental runs and theoretical analysis, we explore the maximum discriminative power that can be expected of a higher order pattern for different numbers of variables. In addition, this work has resulted in a formulation of the pattern generation problem in terms of the selection of rows from a truth table. This formulation is more efficient and provides deeper insight into the process of creating higher order patterns than the formulation obtained from a straightforward translation of the desired pattern characteristics into mathematical constraints. It also makes it easy to define patterns other than just those based on the conjunctive logic used by traditional association and discriminant pattern analysis. For example, it is possible to define a pattern that is present if j out of the k variables have a value of 1. It is also possible to impose additional constraints on the patterns to further tailor them to specific needs.

Overview: In Section 2 we begin with a more formal definition of discriminative patterns and of the *DiffSup* measure that is used to evaluate the discriminative power of such patterns. Section 3 presents the basic approach to pattern specification and generation, while Section 4 describes a more powerful approach that is more efficient and more general. Experimental results are presented in Section 5, and more formally analyzed in Section 6. Section 7 summarizes the paper and the areas for future work.

[1] As measured in terms of the *DiffSup* measure used in this paper, but perhaps not in terms of other measures, such as statistical significance.

[2] By the time pattern sizes of 9 or 10 are reached, computational difficulties arise in some cases.

2 Discriminative Patterns

This section provides a formal description of discriminative patterns and the *DiffSup* measure that is often used to measure the discriminative power of a binary variable or a set of binary variables. This is followed by a brief overview of previous work in discriminative pattern mining. As mentioned, we are not aware of previous work in generating synthetic discriminative patterns. (Again, we are only considering those patterns whose discriminative power is greater than that of any subpattern.)

2.1 Definitions

Let D be a binary transaction data set consisting of m transactions, each of which is a subset of n possible items. D can be represented as a binary data matrix, where each item is a binary variable (column of the matrix), each transaction is a row, and the ij^{th} entry is 1 if transaction i has item j. For instance, the transactions (rows) could be subjects in a medical study, while the binary variables (columns) represent the presence or absence of various genetic features in a subject. In many cases the transactions are divided into classes, e.g., cases (those subjects with a condition) and controls (those without). Without loss of generality, we only consider discriminative patterns for the binary class problem. An extension to multiple classes is described in [2].

Assume there are $m = m_A + m_B$ transactions, where m_A is the number of cases and m_B is the number of controls. Instead of viewing an itemset, X, as a set of items, we will find it more convenient to use an equivalent representation in which the itemset is represented as a logical conjunction of Boolean variables, i.e., as $X = x_{i_1} x_{i_2} ... x_{i_k}$, where x_{i_j} is the Boolean variable associated with the j^{th} item. Let $support\ count_A(X)$ and $support\ count_B(X)$ be the number of transactions for which X is true in classes A and B, respectively. Then, the relative supports of X in classes A and B are defined as $RelSup_A(X) = \frac{support\ count_A(X)}{m_A}$ and $RelSup_B(X) = \frac{support\ count_B(X)}{m_B}$, respectively.

DiffSup, which was originally defined in [2] is the absolute difference of the relative supports of X in classes A and B.

$$DiffSup(X) = |RelSup_A(X) - RelSup_B(X)| \tag{1}$$

An itemset, X, is $r - discriminative$ if $DiffSup(X) \geq r$. The goal of discriminative pattern mining is to discover all patterns in a data set with *DiffSup* greater than or equal to r. However, higher order patterns are not useful or interesting unless their discriminating power is greater than the discriminating power of their subpatterns.

Other measures of 'goodness' for discriminative patterns are sometimes used. For instance, instead of taking the difference of relative support, an alternative measure of discriminative power, the *Growth Rate*, can be defined as the ratio of the two supports [4]. Other variations include information gain [3], the χ^2-test [2], the Gini index [13], the odds ratio [13], and various other measures

[12]. These discriminative measures are generally not anti-monotonic as shown by [4,2,3], a fact that poses significant challenges both for pattern finding and generation. It is possible that our approach could be used to generate higher order patterns for some of these other measures, but we have not yet explored that possibility.

2.2 Previous Work in Discriminative Pattern Mining

Discriminative patterns have been investigated by a number of researchers, sometimes under other names. Dong and Li [4] define *emerging patterns (EP)* as itemsets with a large growth rate (support ratio) between two classes. Emerging patterns have been extended to several special cases such as jumping emerging patterns [9] and minimal emerging patterns[11,10]. In [2], contrast sets (CSETs), were proposed as another possible formulation of discriminative patterns and an algorithm to mine them, CSET, was presented. In [8], contrast set mining was shown to be a special case of rule learning, where the contrast set is the antecedent of a rule whose consequent is a group. As mentioned above, various statistical discriminative measures have also been studied for discriminative pattern mining. There has also been recent work on the efficient discovery of low support discriminative patterns from dense and high-dimensional data sets [6]. Another approach [5] builds a decision tree with frequent patterns at each decision node that partition the data set into successively purer subsets.

3 Defining Higher Order Patterns with Constraints

The starting situation is a set of k binary variables which take on some assignment of zeros and ones (truth values) for a set of m_A cases and m_B controls in a set of $m = m_A + m_B$ subjects. The goal is to find one or more assignments of values to the variables that maximizes the *DiffSup* of the combination of all variables, while ensuring that the *DiffSup* of lower order combinations is less than a specified limit. A more formal definition is provided below.

Problem Statement: *Find an assignment of truth values to m instances (m_A in cases, m_B in controls) of the variables, $x_1, x_2, \ldots, x_k$, that satisfies the following objective and constraints:*

maximize *DiffSup*($x_1 x_2 \ldots x_k$)

subject to *DiffSup*($x_{i_1} x_{i_2} \ldots x_{i_j}$) $\leq$ *limit*, where $1 \leq j < k$ □

This statement of the problem combines constraint satisfaction with optimization. Generating solutions for just the constraint portion of the problem would typically yield size k patterns whose *DiffSup* is less than that of their subpatterns and thus not interesting.

This type of constrained optimization problem, although simply stated, needs to be translated into a practical optimization model. For this purpose, we chose

AMPL (A Modeling Language for Programming) [7]. "AMPL is a computer language for describing production, distribution, blending, scheduling and many other kinds of problems known generally as large-scale optimization or mathematical programming" and has a long history of use in the optimization community. Although we used a commercial implementation of AMPL, along with the IBM ILOG CPLEX Optimizer as a solver, there are non-commercial versions of AMPL (or AMPL subsets) available. For instance, GLPK (GNU Linear Programming Kit), may also provide a suitable platform for solving the problem posed above. However, we have not tried our examples on GLPK.

A key problem in the translation, and indeed, in the solution of this problem is the fact that the satisfaction of the constraints and maximization of the objective depend on the combinations of all variables. This poses at least two major challenges. First, the number of the combinations is $2^k - 1$, where k is the number of individual variables. Thus, the number of constraints grows exponentially with k. However, for pattern sizes of 10 or less (1023 combinations or less) we found this manageable.

The second challenge arises because the values of the combinations are, of course, functions of the variable values being sought. This functional relationship needs to be specified either implicitly or explicitly. When an implicit approach is used, the AMPL model file requires (1) defining a variable for each combination, 7 in all, and (2) defining constraints for each of the combinations. This approach generates a higher order pattern with the desired properties, but the model file rapidly increases in size. Although this file could be generated automatically, we also discovered that the solver was having difficulty producing a solution once we got to five variables. Thus, although this representation provides a relatively direct translation of the problem statement given above, it is not as compact or efficient as the approach described in the next section.

4 A More General Approach

Given the limitations of the previous solution, we sought a new formulation that would provide a more flexible and efficient approach to specifying and generating higher order patterns, as well as providing deeper insight into the nature of higher order patterns and the ability to specify patterns other than those involving logical *and*.

The approach is the following. For k variables, define the truth table that gives the possible values of the variables and all the combinations of variables up to size k. For now, the combinations are assumed to be logical *and*. For example, a truth table for 3 variables and its combinations is shown in Table 2. The same constraint problem is solved, but the constraint solution is transformed into one of selecting, m_A rows for cases and m_B rows for controls, from the 2^k rows of the truth table, so that the constraints given in the problem statement above are satisfied. In this approach, the relationships between the combinations of variables and individual variables are defined explicitly and do not need to be specified as constraints. As a result, the solver has far fewer constraints to handle and runs (in our experience) far more quickly.

Table 2. Truth table for three binary variables and their combinations

row	x_1	x_2	x_3	x_1x_2	x_1x_3	x_2x_3	$x_1x_2x_3$
1	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0
3	0	1	0	0	0	0	0
4	0	1	1	0	0	1	0
5	1	0	0	0	0	0	0
6	1	0	1	0	1	0	0
7	1	1	0	1	0	0	0
8	1	1	1	1	1	1	1

The model file is also much simpler and does not need to be changed as the number of variables changes. Figure 1 shows the new AMPL model file. This file is much smaller than the model file for the original approach (which is not shown in order to save space) and is not specific to any number of variables. This approach is also more flexible than the original one in several ways. First, variable combinations can be defined to be something other than logical *and* without changing the model file. For instance, we could define a combination of size k to be 1 if at least $k-1$ of the variables are 1. This is not to say that any truth table that can be created will be meaningful or even have a feasible solution. Second, we could choose to only use some of the rows of the truth table. Again, this may not yield a feasible solution. However, for example, it is possible to omit the 0 row (i.e., the assignment of zeros to all variables) and still obtain solutions, sometimes with a similar value of the objective function.[3]

Admittedly, this approach also has a number of limitations. The table grows in size exponentially as the number of variables increases. The number of constraints that must be solved by the solver, although fewer than in the previous approach where combinations were defined via constraints, is still exponential in k as well, specifically $2^k - 1$. (There is no escaping that unless the problem statement is changed.) In practice, we found that for a larger number of variables (9 or 10), larger numbers of cases and controls (200 or more), and smaller limits on *DiffSup* (0.3 or less) the solver either ran out of memory (we were using 32 bit AMPL and CPLEX) or took so long to run that we stopped the job. Nonetheless, most of the other cases ran in very little time, i.e., just a few seconds or minutes.

For both approaches, it is possible to add additional constraints. For instance, we used this algorithm plus a few additional constraints to generate the specialized discriminative patterns found by the algorithm described in [6]. This algorithm finds only a subset of discriminative patterns, but can find patterns missed by other discriminative mining algorithms. By specifying and generating these patterns, we gained a better understanding of the types of patterns this algorithm discovers or misses.

[3] However, the row of all zeros does play an important role in creating a data set with the optimal value and often shows up in our solutions.

```
param limit >= 0.0 ;  param mA > 0 ;  param mB > 0 ;

param nv0 >= 2 ;  param nv = (2^nv0) ;  param nvm1 = nv - 1 ;

set CLASSA = 1 .. mA by 1 ;
set CLASSB = (mA+1) .. (mA+mB) by 1 ;

set SINGLES = 1 .. nv0 by 1 ;

set SUBJECT = 1 .. (mA+mB) by 1 ;
set COMBINATION = 1 .. (nv-1) by 1 ;
set COMBINATIONm0 = 1 .. (nv-2) by 1 ;

set VALUES = 1 .. nv by 1 ;
set LINKS = { i in SUBJECT, j in VALUES } ;

param L { i in VALUES, j in COMBINATION } ;
var Select { (i , j) in LINKS } binary ;

maximize diffX0:  (1/mA) * sum { i in CLASSA } ( sum { j in VALUES } Select[ i , j ] * L[ j , nvm1 ] ) -
                  (1/mB) * sum { i in CLASSB } ( sum { j in VALUES } Select[ i , j ] * L[ j , nvm1 ] ) ;

subject to dij { k in COMBINATIONm0 }:
-limit <= (1/mA) * (sum { i in CLASSA } ( sum { j in VALUES } Select[ i , j ] * L[ j , k ] )) -
          (1/mB) * (sum { i in CLASSB } ( sum { j in VALUES } Select[ i , j ] * L[ j , k ] )) <= limit ;

subject to oneperSUB { i in SUBJECT }:  sum{ (i,j) in LINKS } Select[ i , j ] = 1 ;
```

Fig. 1. AMPL model file for the truth table approach. The truth table implicity specifies the relationship between a combination of variables and the individual variables.

5 Experimental Results

This section presents experimental results that show how the optimal *DiffSup* value that is found varies with the *DiffSup* limit for the subpatterns, the number of variables, and the size of the data set, i.e., number of cases and controls. To show this for both balanced and unbalanced data sets, we used six cases. For both unbalanced and balanced data sets, m_A took the values 10, 25, 50, 100, 200, and 250. For unbalanced data sets m_B was double, while for balanced data sets it was, of course, the same. We summarize the results and present a few plots that support our summary.

First, the optimal *DiffSup* value does not vary much with the size of the data set for a fixed limit and number of variables. Intuitively, once the data set is large enough to achieve a certain pattern, this pattern can be repeated multiple times for larger data sets. This is true for both balanced and unbalanced data sets. This is illustrated in Figure 2 for a balanced data set with 5 variables which shows the optimal *DiffSup* values across different *DiffSup* subpattern constraint levels and numbers of cases. Because of this lack of variation, we illustrate the rest of the points we want to make by using data sets with $m_A = 50$.

A minor point is that the optimal *DiffSup* value sometimes decreases slightly as m_A increases from 10 to 25, e.g., for limits 0.3 and 0.5. As we will see in the next section, reducing the *DiffSup* values of subpatterns less than size k while maximizing the *DiffSup* of the size k pattern requires adding a certain number of records in cases and controls to 'balance out' the occurrence of the subpatterns of size less than k. Intuitively, this is more difficult when the data set size is small. Our conjecture is that for 5 variables data sets of size 10 and 25 are both too small to perform this balancing process as well as in larger data sets, but

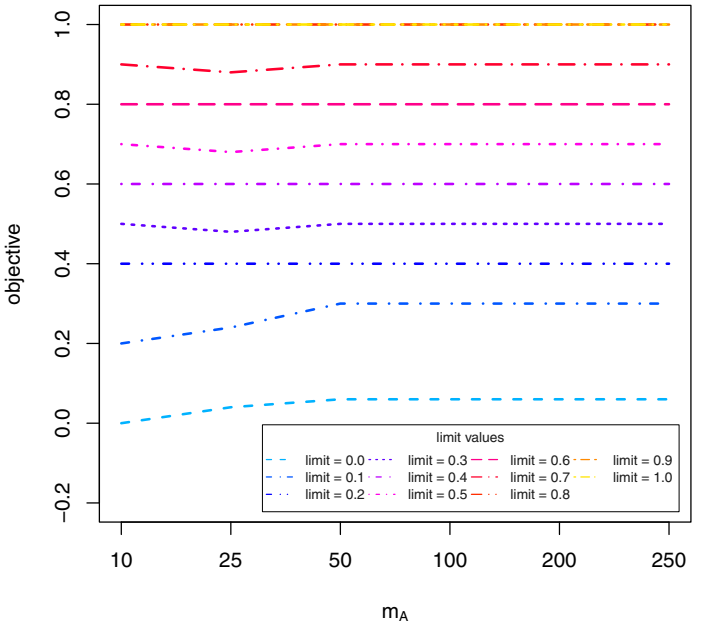


Fig. 2. Optimal *DiffSup* values for 5 variables across different limits and numbers of cases. $m_A = m_B$

that the balancing works more efficiently for a data set of size 10 than for a data set of size 25. However, a formal analysis is needed to confirm this.

The second major observation is that for a given limit and data set size, the optimal *DiffSup* value decreases as the number of variables increases. This is not surprising, since intuitively, there are far more constraints to be satisfied as the number of variables increases. (Recall the number of constraints is $2^k - 1$.) This is shown for both the balanced and the unbalanced cases by figures 3 and 4. These figures also show that the optimum attained in an unbalanced data set is often less than that of the corresponding balanced set, at least for smaller values of the limit. Also note that for both balanced and unbalanced, the *DiffSup* optimum does not change much once 9 or 10 variables are reached. Note that the results for 10 variables are shown as a dashed line, while the results for 9 variables are shown just as orange crosses, without any accompanying line.

6 Formal Analysis

It is possible, at least in some cases, to perform a more formal analysis of this approach to determine the optimal *DiffSup* as the number of variables increases. We briefly sketch this approach for $limit = 0$ and balanced data sets. The approach is to fill out the m rows of the data file one by one. We begin by putting a row consisting of all ones in the cases. Every combination, including that of k variables, has the value 1. To achieve the goal of maximizing the *DiffSup*

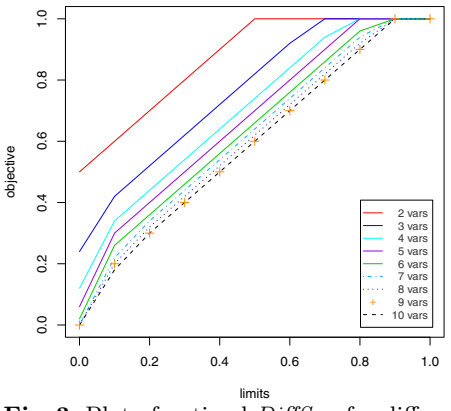


Fig. 3. Plot of optimal *DiffSup* for different limits and number of variables. $m_A = m_B = 50$.

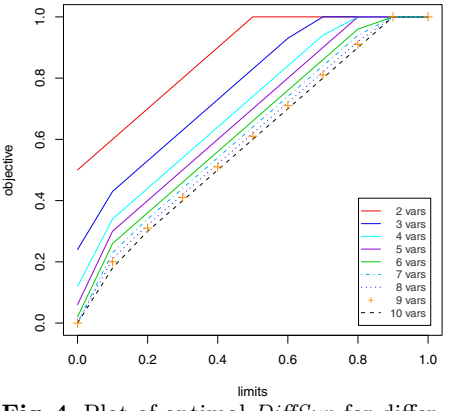


Fig. 4. Plot of optimal *DiffSup* for different limits and number of variables. $m_A = 50, m_B = 100$.

for the size k combination, while maintaining every other combination with a 0 *DiffSup*, it is necessary to add rows to cases and controls that 'cancel out' all other combinations.

We begin by canceling out the $\binom{k}{k-1} = k$ subpatterns of size $k-1$. This can be done by adding, for each such subpattern, the row from the truth table that contains a 1 for that pattern. For example, if $k = 3$, then there are 3 pairs of variables. Rows 4, 6, and 7 of Table 2 are the rows in the truth table that will cancel the net occurrence for these pairs when placed in controls. This accomplishes the goal that the *DiffSup* of the pairs is 0. More generally, canceling out the k subpatterns of size $k-1$ requires that k rows be added to controls.

However, doing this increases the occurrence of the size $k-2$ subpatterns in controls by 2. Each $k-1$ subpattern contains $\binom{k-1}{k-2} = k-1$ subpatterns of size $k-2$. Thus, when $k = 3$, pairs contain two individual subpatterns of size 1 (which is obvious). Since there are $\binom{k}{k-2} = k(k-1)/2$ patterns of size $k-2$, the total occurrence in controls of a $k-2$ size pattern from the k rows just added is $k(k-1)/(k(k-1)/2) = 2$. More generally, adding these k rows increases the occurrence in controls of size $k-i$ patterns, $1 \leq i \leq (k-1)$, by i. (Proof omitted.) For instance, when $k = 3$, pairs have an occurrence in controls of 1 and individual variables have an occurrence of 2.

Thus, some rows need to be added to cases to cancel out the excess count that was added to controls. This can be done by adding $\binom{k}{k-2}$ rows from the truth table, where these rows all have a value of 1 for a particular $k-2$ size pattern, but do not have a 1 for any higher level pairs. While this reduces the *DiffSup* of all the $k-2$ size patterns to 0, size $k-3$ patterns now have a excess count of 1 in cases. This process continues, adding $\binom{k}{k-i}$ patterns of size $k-i$ alternately to cases and control until k rows corresponding to individual variables are added to either cases or controls.

Thus, adding one row of all ones to cases requires adding many rows to both cases and controls to achieve 0 *DiffSup* in the subpatterns. It is possible to

compute how many rows are added to cases and controls and then to compute the value of *DiffSup* for the size k pattern. $m_A = \sum \binom{k}{i}$, $i = k, k-2, \ldots, 2$ or 1 depending on whether k is even or odd, respectively, and $m_B = \sum \binom{k}{i}$, $i = k-1, k-3, \ldots, 1$ or 2 depending on whether k is even or odd, respectively. These values will differ by 1 and thus, to get the *DiffSup* of the patterns to actually cancel will require adding the row consisting of all zeros from the truth table to either cases or controls to make the data set balanced. This represents the best solution (proof omitted) for $limit = 0$ and thus constitutes an upper bound.

Table 3 shows the number of rows this process generates in cases and controls for each variable size (omitting the one zero row that must be added to balance the data set). This table also shows the maximum *DiffSup* attainable. We can make some observations (which can be more formally proven), e.g., that the $m_A = m_B = 2^{k-1}$ and $DiffSup = 1/2^{k-1}$. This agrees with Figure 3 and even appears to hold for the unbalanced data results shown in Figure 4. This result could be used as the basis for a simple algorithm that creates optimal size k patterns for $limit = 0$ and balanced data sets that are multiples of size 2^k.

Table 3. Table showing the m_A and m_B values required for 0 *DiffSup* subpatterns and maximal *DiffSup* for the size k pattern.

num vars	2	3	4	5	6	7	8	9	10
m_A	1	4	7	16	31	64	127	256	511
m_B	2	3	8	15	32	63	128	255	512
DiffSup	0.5	0.25	0.125	0.0625	0.0313	0.0156	0.0078	0.0039	0.0020

Although this analysis is interesting and yields some useful insights, much more analysis is possible and could well yield equally interesting results.

7 Conclusion and Future Work

We have presented a constraint based approach for generating higher order combinations of individual variables which may show high discrimination even when the single variables or lower order combinations little or no discrimination. The basic approach quickly lead to a new approach based on building a data set by choosing rows from a truth table. This formulation provides more insight into the process of creating higher order patterns than does the formulation obtained from a straightforward translation of the desired pattern characteristics into mathematical constraints, as was demonstrated by the derivation of an upper bound for the *DiffSup* of a size k pattern for a balanced data set and $limit = 0$. It also allows for the easy definition of patterns other than just those based on the traditional conjunctive logic used by traditional association and discriminant pattern analysis. Experimental results and formal analysis were presented to give insight into the behavior of higher order patterns.

There are many directions worthy of further investigation. First, there are many aspects of the results presented here that remain to be addressed. One is

establishing bounds on the *DiffSup* of size k patterns for different types of data sets, different numbers of variables, and different limits. We plan to investigate alternative approaches for analyzing the problem that may provide such answers more readily than the two approaches presented in this paper. Another task is to investigate truth tables that reflect types of logic other than logical *and* or that require solutions that don't involve a row of all zeros. Finding larger sets of solutions, even if slightly suboptimal, might also be useful since the goal is to generate different types of discriminative patterns. Along similar lines, we could subject different subpatterns to different constraints. We could also explore other types of solution methods, either other constraint based methods or non-constraint based methods. It is even conceivable that a thorough analysis could produce a non-optimization based algorithm for generating higher order discriminative patterns. It would be especially interesting to investigate if the insights that arise from our work can lead to new or improved discriminative pattern mining algorithms. Yet another significant challenge is to investigate other measures of discrimination .

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB 1994: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499. Morgan Kaufmann Publishers Inc., San Francisco (1994)
2. Bay, S., Pazzani, M.: Detecting group differences: Mining contrast sets. Data Mining and Knowledge Discovery 5(3), 213–246 (2001)
3. Cheng, H., Yan, X., Han, J., Hsu, C.-W.: Discriminative frequent pattern analysis for effective classification. In: Proceedings of International Conference on Data Engineering, pp. 716–725 (2007)
4. Dong, G., Li, J.: Efficient mining of emerging paterns: Discovering trends and differences. In: Proceedings of the 2001 ACM SIGKDD International Conference on Knowledge Discovery in Databases, pp. 43–52 (1999)
5. Fan, W., Zhang, K., Cheng, H., Gao, J., Yan, X., Han, J., Yu, P., Verscheure, O.: Direct mining of discriminative and essential frequent patterns via model-based search tree. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2008, pp. 230–238. ACM, New York (2008)
6. Fang, G., Pandey, G., Wang, W., Gupta, M., Steinbach, M., Kumar, V.: Mining low-support discriminative patterns from dense and high-dimensional data. IEEE Transactions on Knowledge and Data Engineering (2010) (in press)
7. Fourer, R., Gay, D.M., Kernighan, B.W.: A modeling language for mathematical programming. Manage. Sci. 36(5), 519–554 (1990)
8. Geoffrey, D.A.N., Webb, I., Butler, S.M.: On detecting differences between groups. In: Proceeding of the ACM SIGKDD International Conference on Knowledge Discovery in Databases, pp. 256–265 (2003)
9. Li, J., Dong, G., Ramamohanarao, K.: Making use of the most expressive jumping emerging patterns for classification. Knowledge and Information systems 3(2), 131–145 (2001)

10. Li, J., Liu, G., Wong, L.: Mining statistically important equivalence classes and delta-discriminative emerging patterns. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 430–439. ACM, New York (2007)
11. Loekito, E., Bailey, J.: Fast mining of high dimensional expressive contrast patterns using zero-suppressed binary decision diagrams. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 307–316. ACM, New York (2006)
12. Morishita, S., Sese, J.: Transversing itemset lattices with statistical metric pruning. In: Proceedings of the Nineteenth ACM Symposium on Principles of Database Systems, pp. 226–236. ACM, New York (2000)
13. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to data mining. Addison-Wesley, Reading (2005)

Mining Maximal Co-located Event Sets

Jin Soung Yoo and Mark Bow*

Department of Computer Science, Indiana University-Purdue University,
Fort Wayne, Indiana, USA
{yooj,bowmg01}@ipfw.edu

Abstract. A spatial co-location is a set of spatial events being frequently observed together in nearby geographic space. A common framework for mining spatial association patterns employs a level-wised search method (like Apriori). However, the Apriori-based algorithms do not scale well for discovering long co-location patterns in large or dense spatial neighborhoods and can be restricted for only short pattern discovery. To address this problem, we propose an algorithm for finding maximal co-located event sets which concisely represent all co-location patterns. The proposed algorithm generates only most promising candidates, traverses the pattern search space in depth-first manner with an effective pruning scheme, and reduces expensive co-location instance search operations. Our experiment result shows that the proposed algorithm is computationally effective when mining maximal co-locations.

1 Introduction

As one of important spatial data mining tasks, spatial association pattern mining has been popularly studied for discovering spatial dependencies among objects [19,23,25,9,16,13,8,22]. Shekhar et al. [19] defined a spatial association pattern (called *spatial co-location*) based on clique neighbor relationships among spatial features. A spatial co-location pattern represents "a set of spatial features which are frequently observed together in a spatial proximity". Examples of co-location patterns include symbiotic species, e.g., West Nile disease and stagnant water sources in epidemiology [12], and interdependent events, e.g., traffic jam, car accident, ambulances and police men in transportation [18]. Co-location mining results can be also used for finding dependencies among services requested by mobile users close in the geographic sense or discovering the relationship of spatial distributions among various kinds of mineral elements for geologists.

The problem of mining association rules based on spatial predicates (e.g.,*close to*, *near* and *overlap*) was first discussed in [13]. Most of spatial association mining algorithms [16,13,19,23,25] use a generation-and-test method like *Apriori*. Apriori-inspired co-location mining algorithms traverse the search space in a breadth-first manner and find prevalent co-located event sets by enumerating their co-location instances from an input spatial dataset. In order to produce a co-located event set of length k, all 2^k of its subsets are searched since they too must be co-location patterns. The size of search space in spatial co-location

* Corresponding author.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 351–362, 2011.

mining gives more significant effect on the computational performance than classical association mining since a large fraction of the computation time is devoted to identifying co-location instances having clique relationships from input spatial data. When very long patterns present in the data, it is often impractical to generate the entire set of co-location patterns. Current algorithms may be restricted for short length patterns or sparse data. However, long co-location patterns often present with large or dense spatial neighborhoods.

To address this problem, we propose an algorithm for finding maximal co-located event sets which concisely represent all co-location patterns. We follow the definition of *maximal* in classical association rule mining [17]. A co-located event set with a minimum prevalence is *maximal* if and only if it has no super event set that is prevalent. In classical data mining literature, there are many works for discovering maximal frequent itemsets from transaction databases [17,26,14,4,10]. However, it is non trivial to reuse them directly for discovering maximal co-locations since, unlike market-basket data, distinct transactions are not explicit in spatial data. Spatial objects are embedded in a continuous space and share a variety of spatial relationships with each other. The depth and complexity of pattern search space in large and dense spatial databases also make challenges in finding maximal co-location patterns.

The main contributions of this work are to formulate the maximal co-location mining problem, and develop an algorithm (MAXColoc) for efficiently extracting only maximal co-located event sets from large spatial data. The proposed algorithm generates only most promising candidates and traverses the pattern search space in both depth-first and breadth-first manners. The depth-first traversal is used in order to quickly identify maximal co-location patterns. A '*subset pruning by a superset*' strategy is used for further reducing the search space throughout the mining process. We use the breadth-first traversal for pruning all subsets of maximal co-locations. Our schemes eventually reduce the expensive operation to find co-location instances forming cliques. We show that the algorithm is correct and complete in finding maximal co-locations. Experimental results with real data and synthetic data show that the proposed algorithm is effectively to reduce the number of candidates and computationally efficient over a state-of-the-art co-location mining algorithm in finding maximal co-locations.

The remainder of this paper is organized as follows. Section 2 shows the basic concept of co-location pattern mining, our problem statement, and the related work. Section 3 describes the proposed algorithm and analysis. The experimental result is presented in Section 4. Section 5 ends with the summary.

2 Problem Statement and Related Work

We first introduce the basic concepts of spatial co-location mining and then present our problem statement and the related work.

2.1 Basic Concepts of Spatial Co-location Mining

Given a set of spatial event types E, a set of their observed objects S, and a neighbor relationship R over S, a **co-location** X is a subset of spatial event types,

$X \subseteq E$, whose objects frequently form cliques under the neighbor relationship (i.e., are neighbors to each other). The term co-location is interchangeably used with a co-located event set throughout this paper. When the Euclidean metric is used for the neighbor relationship R, two spatial objects are neighbors if the ordinary distance between them is not greater than a given distance threshold. Fig. 1 (a) shows an example spatial dataset which has five spatial event types named with A, B, C, D and E. Each object is represented by its event type and unique id, e.g., A.1. Identified neighbor objects are connected by a line in the figure. A **co-location instance** I of a co-location X is a set of objects, $I \subseteq S$, which includes the objects of all event types in X and forms a clique neighbor relationship. For example, in Fig. 1 (a), {A.1, C.1, E.1} is a co-location instance of {A, C, E} but {A.3, C.3, E.2} is not. We also call the co-location instance the **clique instance**. The prevalence strength of a co-location pattern is often measured by its participation index [19]. The **participation index** $Pi(X)$ of a co-location X is defined as $Pi(X) = \min_{e_i \in X}\{Pr(X, e_i)\}$, where $Pr(X, e_i)$ is the **participation ratio** of event type e_i in a co-location $X = \{e_1, \ldots, e_k\}$, that is the fraction of objects of event e_i in the neighborhood of co-location instances of $X - \{e_i\}$, i.e., $Pr(X, e_i) = \frac{Number\ of\ distinct\ objects\ of\ e_i\ in\ instances\ of\ X}{Number\ of\ objects\ of\ e_i}$. The measure indicates that wherever an event in X is observed, with a probability of at least $Pi(X)$, all other events in X can be observed in its neighborhood. For example, in Fig. 1 (a), there are two instances of a co-location X={A,C,E}, {A.1, C.1, E.1} and {A.4, C.1, E.1}. The participation ratio of event A in X, $Pr(X,$ A) is $\frac{2}{4}$ since only A.1 and A.4 among four objects of event A are involved in the co-location instances. In the same way, $Pr(X,$ C) is $\frac{1}{3}$ and $Pr(X,$ E) is $\frac{1}{2}$. Thus the participation index of X, $Pi(X)$, is $min\{Pr(X,$ A), $Pr(X,$ C), $Pr(X,$ E)$\} = \frac{1}{3}$. All co-located event sets having participation index above a given minimum prevalence threshold are returned as prevalent co-locations.

2.2 Problem Statement

We define a key term and the problem statement for finding maximal co-location patterns.

Definition 1. *If X is a co-located event set which is prevalent but no super event set of X is prevalent, we say that X is a* **maximal co-location**.

Given
1) A set of spatial event types $E = \{e_1, \ldots, e_m\}$
2) A dataset of spatial point objects $S = S_1 \cup \ldots \cup S_m$ where $S_i (1 \leq i \leq m)$ is a set of objects of event type e_i. Each object $o \in S_i$ has a vector information of $<$ event type e_i, object id j, location $(x, y) >$, where $1 \leq j \leq |S_i|$.
3) A spatial neighbor relationship R
4) A minimum prevalent threshold θ
Develop
An algorithm to find all maximal co-locations efficiently in computation.
Constraint
The result set of maximal co-locations is correct and complete.

2.3 Related Work

Spatial association rule mining problem was first discussed in Koperski et al. [13]. Castro et al. [6] proposed a clustering-based map overlay approach for discovering spatial association patterns. Most works on co-location pattern mining have presented different approaches for identifying co-location instances and choosing interest measures. Morimoto [16] discovered frequent neighboring class (i.e., co-located event) sets using a *support count* measure. Shekhar et al. [19] proposed statistically meaningful interest measures for co-location patterns and a join-based co-location mining algorithm. Approaches to reduce expensive join operations used in [19] for finding clique instances were proposed in [23] and [25]. Xiao et al. [22] proposed a density based approach for identifying co-location instances. Zhang et al. [27] enhanced the co-location pattern proposed by [19] and proposed an approach to find spatial star, clique and generic patterns. Al-Naymat [2] proposed the problem of enumeration of maximal cliques for mining spatial co-location patterns. This work can find maximal co-location patterns as ours. However it finds prevalent maximal co-locations after retrieving all maximal cliques. Clique enumeration problem is known as NP-Hard, and is well studied in graph theory [1,15,21]. We more concern to reduce candidate event sets and to find clique instances for only filtered candidates.

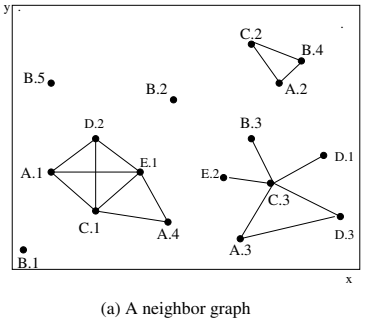
In classical data mining literature, there are many works for discovering maximal frequent itemsets from transaction database [17,26,14,4,10]. Bayardo [17] proposed the Max-Miner algorithm for discovering maximal frequent itemsets. Zaki et al. [26] presented the algorithms MaxEclat and MaxClique for identifying maximal frequent itemsets. Lin et al. [14] have proposed an algorithm called Pincer-Search for mining long maximal frequent itemsets. MAFIA [5,4] is the most recent method for mining the patterns. We adopt one of its pruning strategies for our algorithm.

3 Algorithm

We first describe our algorithmic design concept in four parts: preprocess, candidate generation, candidate pruning and instance filtering, and then present the algorithm. In the end, we analyze the proposed algorithm for completeness and correctness.

3.1 Preprocess

An input spatial dataset can be represented as a neighbor graph with the spatial objects being its vertex set and an undirected edge between two objects where they are neighbors each other as shown in Fig. 1 (a). A brute-force approach to discover maximal co-location patterns is first to find all co-location instances forming cliques from the neighbor graph, compute a participation index per event set and then find out the maximal co-locations. However, it is computationally expensive to find all cliques directly from a graph [7]. Instead, we represent the input spatial data as a set of neighborhood transactions.



(a) A neighbor graph with spatial points and their neighbor relations

Trans No.	Items (Neighbor objects)
1	A.1 , D.2 , E.1
2	A.2 , B.4 , C.2
3	A.3 , C.3 , D.3
4	A.4 , C.1 , E.1
5	B.1
6	B.2
7	B.3 , C.3
8	B.4 , A.2 , C.2
9	B.5
10	C.1 , A.1 , A.4 , D.2 , E.1
11	C.2 , A.2 , B.4
12	C.3 , A.3 , B.3 , D.1 , D.3 , E.2
13	D.1 , C.3
14	D.2 , A.1 , C.1 , E.1
15	D.3 , A.3 , C.3
16	E.1 , A.1 , A.4 , C.1 , D.2
17	E.2 , C.3

(b) Neighborhood transactions

Trans No.	Items (Neighbor events)
1	A , C , D , E
2	A , B , C
3	A , C , D
4	A , C , E
5	B
6	B
7	B , C
8	B , A , C
9	B
10	C , A , D , E
11	C , A , B
12	C , A , B , D , E
13	D , C
14	D , A , C , E
15	D , A , C
16	E , A , C , D
17	E , C

(c) Event neighborhood transactions

Fig. 1. Preprocess: neighborhood transactions

Definition 2. *Given a spatial object $o_i \in S$, the* **neighborhood transaction** *of o_i is defined as a set of spatial objects: $\{o_i, o_j \in S | R(o_i, o_j) = true \wedge o_i's$ event type $\neq o_j's$ event type$\}$, where R is a neighbor relationship.*

For example, in Fig. 1 (a), C.1 has neighbor relationships with each A.1, A.4, D.2 and E.1. The neighborhood transaction of C.1 is {C.1, A.1, A.4, D.2, E.1} including itself as shown in Fig. 1 (b). Note that each object in a transaction has a neighbor relationship with the first object, C.1, which is called a reference object. The set of distinct events in the neighborhood transaction is called an **event neighborhood transaction**. The size of event neighborhood transactions is usually much smaller than the original neighborhood transactions. Event neighborhood transactions are used for generating candidate event sets. Neighborhood transactions are used for filtering their co-location instances.

3.2 Candidate Generation

Rather than considering all possible sets, it would be desirable to focus on only candidates which can have a clique relationship. For example, in Fig. 1 (a), the objects of events A, B and D do not make any clique relationship. If we know this kind of event sets in advance, we can avoid searching their co-location instances. It is important to reduce the number of candidate sets since a large fraction of the computation time is devoted to identifying their co-location instances having clique relationships [24]. The basic idea of our candidate generation is properly combining the event neighbor information of each reference event.

First, we generate event sets for co-location candidates using modified FP-tree and FP-growth algorithm [11]. FP-tree is a popular data structure for association rule mining. We adopt the data structure to store the neighbor relations of a reference event type. We build one FP-tree per each event type and combine the results of frequent event sets from each tree to generate co-location candidates. In Fig. 2, let us see a tree which stores the neighbor relation information of event type A. The tree consists of one root labeled as a reference event type (here, A)

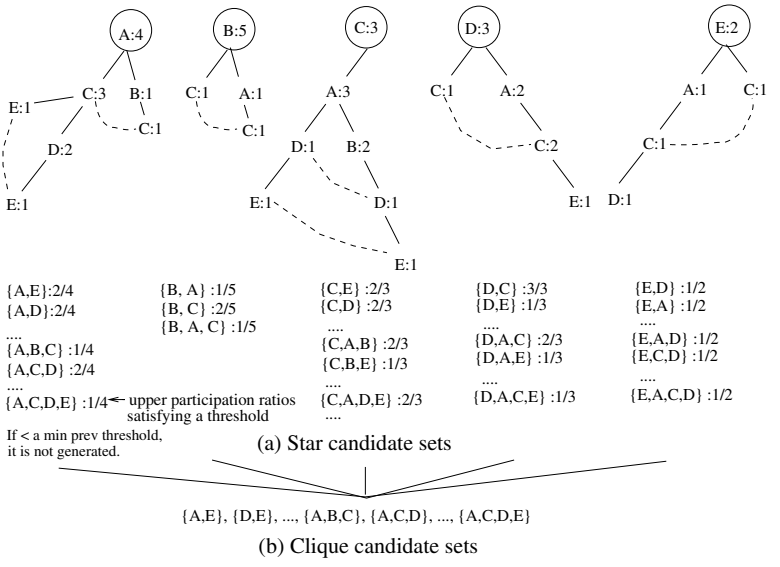
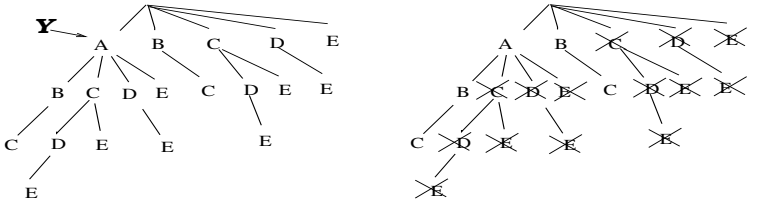


Fig. 2. Candidate generation

and a set of event prefix subtrees as the children of the root. Each node of the tree consists of three fields: $event-type$, $count$, and $node-link$, where $event-type$ denotes an event this node represents, and $count$ registers the number of event neighborhood transactions of the reference event represented by the portion of the path reaching this node. And $node-link$ links to the next node in the tree carrying the same event-type. We call the tree to the **candidate pattern tree** of a reference event.

Frequent event sets with a minimum threshold (i.e., a given prevalence threshold) are generated from each candidate pattern tree using FP-growth algorithm [11]. The difference from the output of the original FP-growth is that each set has the event item of the root node as its first element. Fig. 2 (a) shows event sets generated from each tree. We call the result '**star candidates**' since all elements in a set have neighbor relationships with its first element (which was the root node of the tree). The output also gives a frequency information, i.e., support, which presents the frequency that its first item has a neighbor relationship with all other items in the set. That represents the upper bound of the chance (i.e., participation ratio) that the reference event has a clique relationship with the other events in the set.

After generating all star candidates, we combine them for filtering co-location candidates. For example, in Fig. 2 (a), to be {A,B,C} a co-location candidate, three star candidates {A,B,C}, {B,A,C} and {C,A,B} should be there. Fig. 2 (b) shows all combined candidates which are called '**clique candidates**' or '**co-location candidates**'.



(a) A subset tree with clique candidates

(b) After finding a {A C D E} maximal set

Fig. 3. Subset pruning by a superset

3.3 Candidate Pruning

We present additional scheme (*'subset pruning by a superset'*) to reduce the candidates further during the pattern mining process. Fig. 3 (a) shows the pattern search space with candidates generated using a lexicographic subset tree. We assume that there is a total ordering $\geq_L$ of the event item (e.g., lexicographic). In the subset tree, the size k event sets are ordered lexicographically on each level and all children are associated with the earliest subset in the previous level. The event set identifying each node will be referred to as the node's *head*, while possible extensions of the node are called the *tail*. For example, consider node **Y** in Fig. 3 (a). **Y**'s head is {A} and the tail is the set {B,C,D,E}. The head union tail (HUT) is {A,B,C,D,E}. These terms are borrowed from [5].

We traverses the subset tree in both depth-first and breadth-first manners during the maximal mining process. The depth-first traversal is used in order to quickly identify maximal co-location patterns. The breadth-first traversal is used for the subset pruning by supersets. Once we determine maximal event sets at each level, we do the subset pruning with checking whether the HUT of each node is a subset of a current maximal set. If the HUT is a subset, the subtree whose root is the current node is pruned out. For example, in Fig. 3 (a), suppose {A,C,D,E} is a maximal co-located event set. In the first level, the HUT of a node having A is {A,B,C,D,E}. Since it is not a subset of {A,C,D,E}, we cannot prune the subtree whose root is A. Next, see the node having C. The HUT is {C,D,E} which is subset of the maximal {A,C,D,E}. We thus prune the subtree rooted at C. Fig. 3(b) shows the status of the search space tree after pruning all subsets with a maximal co-location {A,C,D,E}.

3.4 Candidate Instance Filtering

To find co-location instances having clique relationships efficiently, we use a filter-and-refine strategy. The instances of a co-location candidate are gathered with scanning neighborhood transactions whose first item's event type is the same with the first item of the candidate. The instances are called **star instances**.

3.5 Algorithm and Analysis

We present an algorithm for mining MAXimal Co-location patterns (MAXColoc). Algorithm 1 shows the pseudo code.

Preprocess(Step 1-2): Given an input spatial dataset and a neighbor relationship, first find all neighboring object pairs using a geometric method such as plane sweep [3], or a spatial query method using quaternary trees or R-trees [18]. Neighborhood transactions are simply generated with grouping the neighboring objects per each object. Event neighborhood transactions are generated with event types in the neighborhood transactions.

Candidate Generation (Step 3-6): A candidate pattern tree per each event type is constructed with the event neighborhood transactions of the reference event. Star candidates with the minimum prevalence are generated using a project based mining algorithm (FP-growth) [11]. Co-location candidates are filtered with combining the star candidates.

Select size k co-location candidates (Step 7-9): The maximal pattern mining is processed from the longest size of candidate events to size 2. Select size k candidates from the candidate pool.

Gather the star instances of candidates (Step 10): The star instances of candidates are gathered from the neighborhood transactions whose first object's event type is the same with the first item of the candidate.

Filter the co-location instances of a candidate and compute its participation index (Step 11-15): Next filter true co-location instances from the star instances of a candidate examining all neighbor relationships of objects except the first object in the star instance. Here we use neighbor pair information generated in preprocess without additional geographic operations. After finding all true instances of a candidate, compute its participation index.

Update the result set and prune the subsets (Step 16-17): If the candidate set's participation index is greater than a prevalence threshold, it becomes a maximal co-location and is inserted in the result. All subsets of the maximal set are removed from the set of remaining candidates.

Return the final result (Step 20): The procedure of step 8 to step 19 is repeated until k reaches to 2 or there is no candidate. Finally, return the final result of the maximal patterns.

Next we analyze the proposed algorithm (MAXColoc) for completeness and correctness. Completeness means MAXColoc finds all maximal co-locations which satisfies Definition 1. Correctness means that co-located event sets discovered by MAXColoc are all maximal co-locations.

Theorem 1. *The MAXColoc algorithm is complete and correct.*

Proof. The completeness can be briefly explained in the following two parts. First, we show that the candidate generation and pruning procedures do not drop any potential maximal event sets. The candidate generation procedure (*step 3-6*)

Algorithm 1. MAXColoc algorithm

Inputs
$E = \{e_1, \ldots, e_m\}$:a set of spatial event types
S:a spatial dataset, R:a spatial neighbor relationship
θ:a minimum prevalence threshold

Variables
NT: a set of neighborhood transactions
ENT: a set of event neighborhood transactions
NP: a set of all neighbor pairs (size 2)
$Tree_i$: the candidate pattern tree of type e_i, k:interest co-location size
C:a set of all candidate sets, C_k:a set of size k candidates
pi: participation index
CI_c: a set of clique instances of a candidate c
SI_c: a set of star instances of a candidate c
SI_k: a set of star instances of size k co-located event sets $SI_c \in SI_k$
R:a set of maximal co-located patterns
R_k:a set of size k maximal co-located patterns

MAXColoc
1) NP=find_neighbor_pairs(S, R);
2) (NT,ENT)=gen_neighbor_transactions(NP);
3) for i=1 to m do
4) $Tree_i$=build_candidate_pattern_tree(e_i, ENT, θ);
5) end do
6) C=gen_candidates($Tree_1, \ldots, Tree_m$);
7) k = Find_longest_size(C);
8) while($k \geq 2$ or $C \neq \emptyset$) do
9) C_k = Get_k_candidates(C, k);
10) SI_k = Find_star_instances(C_k, NT);
11) for each candidate c in C_k do
12) CI_c = Find_clique_instances(SI_k, c, NP);
13) pi_c = Calculate_pi(CI_c);
14) if pi_c > θ then Insert(c, R_k);
15) end do
16) R=$R \cup R_k$; C=C-C_k;
17) C=Subset_Pruning(R_k, C);
18) k=k-1;
19) end do
20) return R;

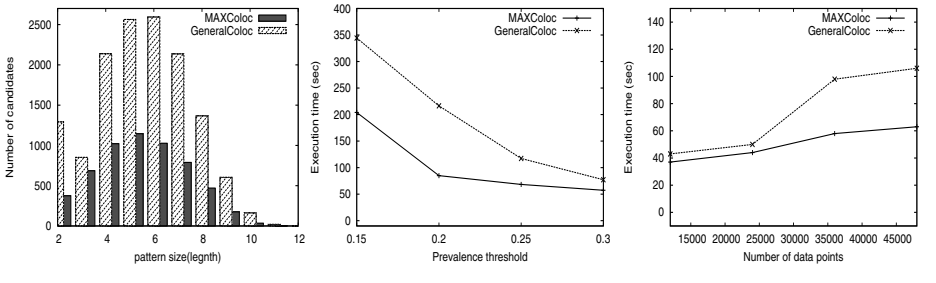
filters a candidate event set using the upper bound of participation ratio of its reference event and the combine procedure is correct. The true participation ratio of an event is not greater than its upper bound value. The participation index of a co-location is a minimum value of all participation ratios of it. The subset pruning procedure (*step 17*) drops only the subsets of maximal co-locations since the comparision with the HUT of a node is correct. Second, we need to show the methods to find co-location instances are correct. The neighborhood transactions generated from the neighboring objects of each object do not miss any neighbor relation (*step 1-2*). Instances gathered from the neighborhood transactions whose first item's event type is the same as the first item of a co-located set, has correct star relationships(*step 10*). Any co-location instance is not missed since the star

instances are a super set of the clique instances of co-location. It is also correct to filter co-location(clique) instances from the star instances using neighbor pair information (*step 12*). In the other hand, the correctness of MAXColoc algorithm can be guaranteed by *steps 14* and *16*. By *step 17*, the candidate pool keeps maximal candidates. If a candidate is prevalent, it becomes a maximal set.

4 Experimental Results

We examined the computational effectiveness of MAXColoc in mining maximal co-location patterns. We compared MAXColoc with a general co-location mining algorithm [19] which is called 'GeneralColoc' in this paper. GeneralColoc has a postprocess to return only maximal sets. We used synthetic data and real data for the experiment. A synthetic dataset(DATASET#1) had 54 event types and a total of 5378 data points. The max neighbor degree (number of neighbors) is 31. The real data is about points of interest(POI) in California from [20]. We view the category type such as 'church', 'school' and 'cliff', as event type . We made several experiment datasets from this base data. One dataset (DATASET#2) has 9949 data points in San Francisco area with 50 different event types. The other datasets (DATASET#3, #4, #5 and #6) have each 12000, 24000, 36000 and 48000 data points with 40 event types. All the experiments were performed on a PC Linux system with 2.0 GB main memory. Main programming language for the implementation is C++.

1) Comparison in the number of candidates: First, we examined the numbers of candidates considered for finding their co-location instances from MAXColoc and GeneralColoc using DATASET#1. The minimum prevalence threshold was 0.1. As shown in Fig. 4 (a), MAXColoc generates much less number of candidates compared with GeneralColoc. MAXColoc has zero candidate at size 2 and the mining process finishes at size 3. In contrast, GeneralColoc dramatically increases the number of candidates until size 7.



(a) (b) (c)

Fig. 4. Experimental result

2) Effect of prevalence threshold: Next, we examined the performance effect with different prevalence thresholds. We used DATASET#2 for this experiment. As shown in Fig. 4 (b), the execution times for the MAXColoc and GeneralColoc have decreased with increase of the threshold value. MAXColoc shows overall better performance than GeneralColoc.

3) Effect of the number of data points: In the last experiment, we compared the effect of the number of data points in the different algorithms. We use four datasets different in size (DATASET#3, #4, #5 and #6). As shown in Fig. 4 (c), the execution times increased with number of data points. MAXColoc slowly increased compared with GeneralColoc.

5 Conclusion

In this paper, we propose an algorithm for finding maximal co-located event sets which concisely represents all co-location patterns. The proposed algorithm has a preprocess to convert input spatial data to sets of neighborhood transactions. It reduces examined candidates and traverses the maximal search space in a depth-first manner with an effective pruning mechanism. The algorithm also uses a filter-and-refine strategy for finding co-location instances. A few experiment results show that the proposed algorithm performs more efficiently against a general co-location mining algorithm in finding maximal co-location patterns.

Acknowledgments

This research was partly supported by Information Analytic and Visualization Center, IPFW.

References

1. Akkoyunlu, E.A.: The Enumeration of Maximal Cliques of Large Graphs. SIAM Journal of Computing 2(1), 1–6 (1973)
2. Al-Naymat, G.: Enumeration of Maximal Clique for Mining Spatial Co-location Patterns. In: Proc. of IEEE/ACS International Conference on Computer Systems and Applications (2008)
3. Berg, M., Kreveld, D., Overmars, M., Schwarzkopf, O.: Computational Geometry. Springer, Heidelberg (2000)
4. Burdick, D., Calimlim, M., Flannick, J., Gehrke, J., Yiu, T.: MAFIA: A Maximal Frequent Itemset Algorithm. IEEE Transaction on Knowledge and Data Engineering 17, 1490–1504 (2005)
5. Burdick, D., Calimlim, M., Gehrke, J.: MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases. In: Proc. of ACM-SIGMOD International Conference on Management of Data (1998)
6. Castro, V.E., Murray, A.: Discovering Associations in Spatial Data - an Efficient Medoid based Approach. In: Wu, X., Kotagiri, R., Korb, K.B. (eds.) PAKDD 1998. LNCS, vol. 1394. Springer, Heidelberg (1998)
7. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms. McGraw-Hill Science, New York (2003)

8. Ding, W., Jiamthapthaksin, R., Parmar, R., Jiang, D., Stepinski, T.F., Eick, C.F.: Towards Region Discovery in Spatial Datasets. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 88–99. Springer, Heidelberg (2008)
9. Eick, C.F., Parmar, R., Ding, W., Stepinski, T.F., Nicot, J.: Finding Regional Co-location Patterns for Sets of Continuous Variables in Spatial Datasets. In: Proc. of ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (2008)
10. Gouda, K., Zaki, M.J.: Efficiently Mining Maximal Frequent Itemsets. In: Proc. of ACM-SIGMOD International Conference on Management of Data (1998)
11. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns Without Candidate Generation. In: Proc. of ACM SIGMOD Conference on Management of Data (2000)
12. Huang, Y., Shekhar, S., Xiong, H.: Discovering Co-location Patterns from Spatial Datasets: A General Approach. IEEE Transactions on Knowledge and Data Engineering 16(12), 1472–1485 (2004)
13. Koperski, K., Han, J.: Discovery of Spatial Association Rules in Geographic Information Databases. In: Proc. of International Symposium on Large Spatial Databases (1995)
14. Lin, D.-I., Kedem, Z.M.: Pincer Search: A New Algorithm for Discovering the Maximum Frequent Set. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, p. 105. Springer, Heidelberg (1998)
15. Loukakis, E., Tsouros, C.: A Depth First Search Algorithm to Generate the Family of Maximal Independent Sets of a Graph Lexicographically. Computing 27, 349–366 (1981)
16. Morimoto, Y.: Mining Frequent Neighboring Class Sets in Spatial Databases. In: Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2001)
17. Bayardo Jr., R.J.: Efficiently Mining Long Patterns from Databases. In: Proc. of ACM-SIGMOD International Conference on Management of Data (1998)
18. Shekhar, S., Chawla, S.: Spatial Databases: A Tour. Prentice-Hall, Englewood Cliffs (2003)
19. Shekhar, S., Huang, Y.: Co-location Rules Mining: A Summary of Results. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, p. 236. Springer, Heidelberg (2001)
20. U.S. Geological Survey, http://www.usgs.gov/
21. Tomita, E., Tanaka, A., Takahashi, H.: The Worst-Case Time Complexity for Generating All Maximal Cliques and Computational Experiments. Theoretical Computer Science 363, 28–42 (2006)
22. Xiao, X., Xie, X., Luo, Q., Ma, W.: Density based Co-location Pattern Discovery. In: Proc. of ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (2008)
23. Yoo, J.S., Shekhar, S.: A Partial Join Approach for Mining Co-location Patterns. In: Proc. of ACM ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (2004)
24. Yoo, J.S., Shekhar, S.: A Join-less Approach for Spatial Co-location Mining: A Summary of Results. In: Proc. of IEEE International Conference on Data Mining (2005)
25. Yoo, J.S., Shekhar, S.: A Join-less Approach for Mining Spatial Co-location Patterns. IEEE Transactions on Knowledge and Data Engineering 18(10), 1323–1337 (2006)
26. Zaki, M., Parthasarathy, S., Ogihara, M., Li, W.: New Algorithms for Fast Discovery of Association Rules. In: Proc. of International Conference on Knowledge Discovery in Databases and Data Mining (1997)
27. Zhang, X., Mamoulis, N., Cheung, D., Shou, Y.: Fast Mining of Spatial Collocations. In: Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2004)

Pattern Mining for a Two-Stage Information Filtering System

Xujuan Zhou[1], Yuefeng Li[1], Peter Bruza[1], Yue Xu[1], and Raymond Y.K. Lau[2]

[1] Faculty of Information Technology, Queensland University of Technology, Brisbane, QLD 4001, Australia
{x.zhou,y2.li,p.bruza,yue.xu}@qut.edu.au
[2] Department of Information Systems, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
raylau@cityu.edu.hk

Abstract. As information available over computer networks is growing exponentially, searching for useful information becomes increasingly more difficult. Accordingly, developing an effective information filtering mechanism is becoming very important to alleviate the problem of information overload. Information filtering systems often employ user profiles to represent users' information needs so as to determine the relevance of documents from an incoming data stream. This paper presents a novel two-stage information filtering model which combines the merits of term-based and pattern-based approaches to effectively filter sheer volume of information. In particular, the first filtering stage is supported by a novel rough analysis model which efficiently removes a large number of irrelevant documents, thereby addressing the overload problem. The second filtering stage is empowered by a semantically rich pattern taxonomy mining model which effectively fetches incoming documents according to the specific information needs of a user, thereby addressing the mismatch problem. The experimental results based on the RCV1 corpus show that the proposed two-stage filtering model significantly outperforms both the term-based and pattern-based information filtering models.

Keywords: Pattern mining, Information filtering, User profile, Threshold.

1 Introduction

An Information Filtering (IF) [1] system monitors an incoming document stream to find the documents that match information needs of users. With information filtering, the representation of the user information needs is variously referred to as user profiles or a topic profile where the filters are applied to the dynamic streams of incoming data. Unlike the traditional search query, user profiles are persistent, and tend to reflect a long-term information need [1].

The traditional IF systems make the decision of rejection or reception for a document when it arrives in the stream. The relevant document is displayed to its users without further scrutiny. This decision-making is completed in one

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 363–374, 2011.

step. Such systems often have difficulties in dealing with issues, such as the feature selection on how to remove noisy and non-relevant features, and threshold setting (how to learn the optimal threshold). To improve the robustness of the IF systems, this paper will propose a novel two-stage framework for information filtering.

To illustrate the two-stage IF model, consider an example that may occur in a TV series. Louisa is a girl from a big city looking for a partner. There are three strategies Louisa may adopt, (i) set herself criteria, check out the information of as many people as possible. When Mr Right meets the criteria, she chooses him and lives happily ever after, (ii) date with everyone that is available, rank them according to suitability, then choose the highest-ranked person and live happily ever after, (iii) set herself criteria for rejection, date with a small number of people then choose the best fit.

The first approach has some obvious setbacks. If the criteria are set too high, Louisa may never find Mr Right. If the criteria are too low, she would have difficulties choosing Mr Right. The second strategy is also not practical. What is the point in wasting the energy with every one and ranking the obviously not suitable people? The third approach is a two-stage method. It is the only sensible and efficient way in the three approaches.

Within the new two-stage IF framework, the first filtering stage is supported by a novel rough analysis model which efficiently removes a large number of irrelevant documents. The intention after the first stage is that only a relatively small amount of potentially highly relevant documents remain as the input to the second stage. The second filtering stage is empowered by a semantically rich pattern taxonomy mining model which effectively rank incoming documents according to the specific information needs of a user and fetches the top ranking documents for a user. The initial idea already proposed in the paper [2]. The main contribution of this research work is that a novel rough sets based optimal filtering threshold calibration method has been developed. It was found that a good "balance" must be found between reducing the "noise" at the first stage, and at the same time, effectively matching incoming documents with a semantically rich user profile at the second stage. With the help of the first topic filtering stage, pattern mining and matching can be conducted efficiently and applied to realistic IF settings.

The remainder of the paper is organized as follows. Section 2 highlights previous research in related areas. Section 3 introduces the Rough Set Decision Rule-based Topic Filtering. Section 4 presents filtering model based on the pattern taxonomy mining. The empirical results are reported in Section 5. Section 6 describes the findings of the experiments and discusses the results. Concluding remarks are sketched in Section 7.

2 Related Work

The term-based approaches for IF have been proposed to address the problems of overload and mismatch over the past decades. The term-based IF systems

used terms to represent the user profiles. Such profiles are the most simplest and common representation of the profiles. For examples: the probabilistic models [3], BM25 [4], rough set-base models [5], and ranking SVM [6] based filtering models used the term-based user profiles. The advantage of term-based model is efficient computational performance as well as mature theories for term weighting, which have emerged over the last couple of decades from the IR and machine learning communities. However, term-based models suffer from the problems, such as, the relationship among the words can not be reflected and also, only considering single words as features is the semantic ambiguity.

In the presence of these setbacks, sequential closed patterns used in data mining community have turned out to be a promising alternative to phrases [7]. Pattern mining techniques can be used to find various text patterns, such as co-occurring terms and multiple grams, maximal frequent patterns, and closed patterns, for building up a representation with these new types of features. In [8], data mining techniques have been used for text analysis by extracting co-occurring terms as descriptive phrases from document collections. However, the effectiveness of the text mining systems using phrases as text representation showed no significant improvement. Mining maximal frequent patterns [9] was also proposed to reduce the time complexity of mining all frequent patterns, where an itemset (or a pattern) was maximal frequent if it had no superset that was frequent.

To consider the very important semantic relationships between the terms, a pattern taxonomy model (PTM) for IF has been proposed in [10]. Pattern taxonomy is a tree-like hierarchy that reserves the sub-sequence (that is, "is-a") relationship between the discovered sequential patterns. These pattern based approaches have shown encouraging improvements on effectiveness, but at the expense of computational efficiency. Another challenging issue for PTM is to deal with low frequency patterns because the measures used in data mining to learn profiles turn out to be not suitable in the filtering tasks.

3 Rough Set-Based Topic Filtering

To deal with the uncertainty issues, a Rough Set-based IF model(RSIF) has been developed in [11]. There are two key tasks in developing a RSIF model. The first one is using discovered rough patterns to represent the topic profiles. The second task is deciding an optimal threshold based on the obtained topic profiles. In this paper, only the positive documents will be used to represent the user profile as a rough set. It is less efficient to use the features of the non-relevant documents for an information filter since coverage of the feature descriptions for negative documents can be very large.

3.1 Discovery of R-Patterns

A set of terms is referred to as a *termset*. Given a positive document d_i and a term t, $tf(d_i, t)$ is defined as the number of occurrences of t in d_i. A set of term frequency pairs

$$\hat{d}_i = \{(t, f)|t \in T, f = tf(t, d_i)\}$$

is referred to as an initial *r-pattern* (rough pattern) in this paper.

Let $termset(p) = \{t|(t, f) \in p\}$ be the *termset* of r-pattern p. In this paper, r-pattern p_1 equals to r-pattern p_2 if and only if $termset(p_1) = termset(p_2)$. Two initial r-patterns can be composed if they have the same *termset*. In this paper, we use the composition operation, $\oplus$, that defined in [11] to compose r-patterns. For example,

$$\{(t_1, 2), (t_2, 5)\} \oplus \{(t_1, 1), (t_2, 3)\} = \{(t_1, 3), (t_2, 8)\}$$

(Notice: $\oplus$ is also suitable for patterns with different termsets, e.g., $\{(t_1, 2), (t_2, 5)\} \oplus \{(t_1, 1)\} = \{(t_1, 3), (t_2, 5)\}$).

Based on the above definitions, for a given set of positive documents $D^+ = \{d_1, d_2, \ldots, d_n\}$, there are n corresponding initial r-patterns $\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_n$. We can also group the initial r-patterns that have the same termset into clusters and use their composition, a r-pattern, to represent the cluster. Therefore, the training set of positive documents, D^+, is described as a set of r-patterns, $RP = \{p_1, p_2, \ldots, p_r\}$, where $r \leq n$, and $n = |D^+|$ is the number of positive documents in D. We write this process as $RP = \{p_1, p_2, \ldots, p_r\} = \oplus(\{\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_n\})$.

Let $cluster(p_i)$ be the set of documents (initial patterns) that are composed to generate p_i. We can define the support of a r-pattern p_i as follows:

$$support(p_i) = \frac{|cluster(p_i)|}{|D^+|}. \tag{1}$$

Theorem 3.11 *Let $RP = \{p_1, p_2, \ldots, p_r\}$ be the set of r-patterns discovered in D^+. We have*

$$\sum_{p_i \in RP} support(p_i) = 1.$$

Proof. For any two r-patterns p_i and p_j, we have $cluster(p_i) \cap cluster(p_j) = \emptyset$ since the documents in the different r-patterns have the different termset. Therefore, we have

$$|cluster(p_i)| + |cluster(p_j)| = |cluster(p_i) \cup cluster(p_j)|.$$

Based on this equation and Eq. 1, we also have

$$\sum_{p_i \in RP} support(p_i) = \sum_{p_i \in RP} \frac{|cluster(p_i)|}{|D^+|} =$$

$$\frac{1}{|D^+|} \times |\bigcup_{p_i \in RP} cluster(p_i)| = \frac{1}{|D^+|} \times |D^+| = 1.$$

3.2 Rough Threshold Model

Up to now, the positive documents in the training set have been represented as r-patterns. In the topic filtering stage, discovered r-patterns are employed to filter out most irrelevant documents rather than to identify relevant documents.

Formally the relationship between r-patterns and terms can be described as the following *association mapping* if we consider term frequencies:

$$\beta : RP \rightarrow 2^{T \times [0,1]}, \tag{2}$$

such that

$$\beta(p_i) = \{(t_1, w_1), (t_2, w_2), \ldots, (t_k, w_k)\},$$

where $p_i \in RP$ is a r-pattern; and $w_i = \frac{f_i}{\sum_{j=1}^{k} f_j}$ if we assume

$$p_i = \{(t_1, f_1), (t_2, f_2), \ldots, (t_k, f_k)\}.$$

We call $\beta(p_i)$ the *normal form* of r-pattern p_i in this paper. The association mapping β can derive a function for the weight distribution of terms on T in order to show the importance of terms in the positive documents, which satisfies:

$$pr_\beta(t) = \sum_{p_i \in RP, (t,w) \in \beta(p_i)} support(p_i) \times w \tag{3}$$

for all $t \in T$.

Theorem 3.21 *Let RP be the set of discovered r-patterns, then pr_β is a probability function on T if $\beta(p_i)$ be the* normal form *of all r-pattern $p_i \in RP$.*

Proof. Based on Eq. 3 and Theorem 3.11, we have

$$\sum_{t \in T} pr_\beta(t) = \sum_{t \in T} \sum_{p_i \in RP, (t,w) \in \beta(p_i)} support(p_i) \times w =$$

$$\sum_{p_i \in RP} \sum_{(t,w) \in \beta(p_i)} support(p_i) \times w =$$

$$\sum_{p_i \in RP} (support(p_i) \times \sum_{(t,w) \in \beta(p_i)} w) =$$

$$\sum_{p_i \in RP} support(p_i) \times 1 = 1.$$

Based on the above discussion, a positive document d_i can be described as an event that represents what users want with the probability value. Therefore, the weight of a positive document d_i is

$$W_{d_i} = prob(d_i) = \sum_{t \in d_i \cap T} pr_\beta(t).$$

To work out the suitable thresholds, it is assumed that document d is irrelevant if it is not closed to the common feature of the topic profiles in the training set. For a given topic, it consists of a set of the positive document, D^+. Each document d_i has a weight W_{d_i}. To capture the common feature of the topic from the training data, the distributions of the document weights for a given topic must be first understood.

Many simplistic models assume normal distribution, that is, the data is symmetric about the mean. The normal distribution has a skewness of zero. It is reasonable to assume that the scores of the document follow a normally distributed pattern. Using the mean of the Rough Set weights as a threshold would be a good initial choice, because the mean represents the "common feature". According to the statistical approach, if the distributions of the weights of the documents is assumed as a normal distribution then the common feature, ξ_j for a topic can be modelled as:

$$\xi_j = \frac{1}{n} \sum_{d_i \in D^+} prob(d_i);$$

where n is the number of the positive documents, $n = |D^+|$. In fact, ξ_j is the mean, $\overline{m}$, of the probabilities of the positive documents in D^+. The thresholds, therefore, can be simply determined as $threshold = \xi_j$.

However, real data points are not always perfectly symmetric. Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable. By observations from the experiments conducted in this study, the distributions of the weights of the documents have exhibited a high degree of skewness. To obtain the "real" common feature, both the standard derivation and the skewness must be taken into consideration for modeling the document weights. The following features have been used to characterize a histogram in this paper.

σ is the standard deviation of the probabilities of positive documents. It is given by:

$$\sigma = \sqrt{\frac{1}{n} \sum_{d_i \in D^+} (prob(d_i) - \overline{m})^2} \tag{4}$$

μ is the skewness of the probabilities. The skewness is given by:

$$\mu = \frac{\sqrt{n} \sum_{d_i \in D^+} (prob(d_i) - \overline{m})^3}{(\sum_{d_i \in D^+} (prob(d_i) - \overline{m})^2)^{\frac{3}{2}}} \tag{5}$$

A linear discriminated function is used to make a decision based on features obtained from the above analysis. Therefore, the threshold can be determined as follows:

$$threshold = \xi_j + \gamma(\sigma + \mu) \tag{6}$$

where γ is an experimental coefficient obtained from specific data sets. It is an empirical value. When the user profiles are specific, a lower value of γ can be used and allow more documents into the second stage filtering. Likewise, a

higher value of γ will potentially limit the irrelevant documents into next stage. If adaptive filtering will be the study, γ can be a dynamic parameter (reducing) as the user profiles become more certain.

4 Pattern Taxonomy Mining

In the second stage, all the documents (training and testing documents) are split into paragraphs and each paragraph is treated as an individual transaction (a sequence). Each transaction consists of a set of words (terms). So, a given document d_i yields a set of paragraphs DP. Given a document $d = \{S_1, S_2, ..., S_n\}$, where S_i is a sequence representing a paragraph in d. Let $T = \{t_1, t_2, ..., t_n\}$ be a set of all items - an item is a term which can be a word or keyword in a text document. An itemset is a termset. A sequence S is an ordered list of terms, such that $S = \langle s_1, s_2, ...s_l \rangle$ where s_j is a termset and $s_j \in T$. S is called a sequential pattern of d if there is a $S_i \in d$ such that $S \sqsubseteq S_i$.

Given a termset X in a document $d = \{S_1, S_2, ..., S_n\}$, $\hat{X}$ is used to denote the covering set of X, which includes all sequences $S \in d$ such that $X \subseteq S$, i.e., $\hat{X} = \{S | S \in d, X \subseteq S\}$. The absolute support of X is used to indicate the number of occurrences of X in d. It is denoted as $supp_a(X) = |\{S | S \in d, X \sqsubseteq S\}|$. The relative support of X is the fraction of the sequences that contain the termset X, denoted as $supp_r(X) = \frac{|\hat{X}|}{|d|}$. The purpose of using the relative support of termset X is to properly estimate the significance of the pattern. The absolute support of X only measures the frequency of X in d. A termset X with the same frequency will acquire the same support in documents of varying lengths. However, with the same frequency, a termset X is more significant in a short document than in a long one, because we decompose a document into a set of transactions and discover the frequent patterns in them by using data mining methods; the relative support is estimated by dividing the absolute support by the number transactions in a document. Therefore, a termset X can have an adequate support in various document lengths with the same frequency.

A termset X is called a frequent sequential pattern if its relative support $supp_r(X)$ is greater than or equal to a predefined minimum support, that is, $supp_r(X) \geq min\_sup$. The purpose of using min_sup in our approach is to reduce the number of patterns discovered in a large document. Otherwise, these patterns with a lower relative support will increase the burden of the training. Removing the less significant patterns will save much computational time without affecting the performance very much.

A frequent sequential pattern X is called a closed sequential pattern if there exists no frequent sequential pattern X' such that $X \subset X'$ and $supp_a(X) = supp_a(X')$. The relation $\subset$ represents the strict part of the subsequence relation $\sqsubseteq$. The closed pattern mining can substantially reduce the redundant patterns and increase both efficiency and effectiveness.

The closed sequential patterns discovered from a document collection (a training set) then can be structured into a taxonomy by using the "is-a" relationship between the patterns. Pattern taxonomy is a tree-like hierarchy that reserves the

sub-sequence relationship between discovered sequential patterns [10]. Pattern taxonomy model (PTM) is a rich semantic representation of users information needs. On the other hand, documents are also represented by frequent sequential patterns. The monotonic reasoning framework is then used to infer if certain documents satisfy the information needs of a specific information seeker.

After the pattern taxonomies are obtained, the next step is to develop a method to measure how important patterns are to a given topic, that is, develop a pattern weighting function.

If each mined sequential pattern p in a pattern taxonomy is viewed as a rule $p \rightarrow positive$, then a value can be assigned to each pattern by exploiting the support or confidence of the patterns within the relevant documents. One of existing pattern weighting functions using the confidence of pattern in [12] is as follow:

$$w_{(p)} = \frac{|\{d_i | d_i \in D^+, p \subseteq d_i\}|}{|\{d_j | d_j \in D, p \subseteq d_j\}|}$$

where D is the training set of documents, and D^+ is the set of positive document in D.

However, using support or confidence as a significant measurement of the patterns suffers from time-consuming and ineffective problems. The measures used by data mining ("support" and "confidences") to learn the profile lead to the low frequency patterns problem; thus, it is not suitable in the filtering stage. Patterns have different lengths. Intuitively, the long patterns have more specificity and a lower frequency; the short ones are more general and have a higher frequency. By way of illustration, given a specified topic, a highly frequent pattern (normally a short pattern with a large support) is usually a general pattern or a specific pattern having a low frequency.

Instead of evaluating the pattern's support, we evaluate a term's support (weights) within a discovered pattern and then calculate a specificity value for each pattern. The difference between this term weight method and the IR term-based approaches is that a component of a given term's weighting of the term-based method is based on its appearance in the documents.

Formally, for all positive document $d_i \in D^+$, we first deploy its closed patterns on a common set of terms T in order to obtain the following r-patterns:

$$\overrightarrow{d_i} =< (t_{i_1}, n_{i_1}), (t_{i_2}, n_{i_2}), \ldots, (t_{i_m}, n_{i_m}) > \tag{7}$$

where t_{i_j} in pair (t_{i_j}, n_{i_j}) denotes a single term and n_{i_j} is its *support* in d_i which is the number of closed patterns that contain t_{i_j}.

These r-patterns are composed by using an association mapping (see Eq 2). The supports of terms can be calculated by using Eq 3.

After the supports of terms have been computed from the training set, the specific value of a pattern p for the given topic is defined as follows:

$$spe(p) = \sum_{t \in p} support(t).$$

It is also easy to verify $spe(p_1) \leq spe(p_2)$ if $p_1 \sqsubseteq p_2$, that is, p_1 a sub-pattern of pattern p_2. This property shows that a document should be assigned a large

weight if it contains many large patterns. Based on this observation, we will assign the following weight to a document d for ranking documents in the second stage:

$$weight(d) = \sum_{t \in T} support(t)\tau(t, d).$$

5 Experiments

The Reuters Corpus Volume 1 (RCV1) has been selected to test the effectiveness of the new two-stage information filtering model. All 100 TREC-11 topics have been used in our experiments. $F_1 = \frac{2PR}{(P+R)}$ matrix is used and the paired two-tailed t-test is applied over all 100 topics on F_1 scores.

The proposed two-stage filtering model (T-SM) integrates two types of filtering models: a term-based and a pattern mining-based model. In the first stage, a threshold setting method is developed based on the rough set analysis. In the second stage, a document ranking model is developed based on the pattern taxonomy model. The major objectives of the experiments are to show how the rough threshold model of the first stage can affect the performance of the two-stage filtering system and how the pattern mining method can help improve the performance in the second stage. Hence, to give a comprehensive investigation for the proposed model, our experiments involve comparing the filtering performance of the different threshold setting methods and the different combinations of term-based and pattern-based filtering models.

The Different Threshold Setting Methods. The two-stage models use $threshold_{min}$ developed in [11] and $threshold$ (see Eq. 6) newly developed in this study at the first stage, respectively. The results on F_1 matrix and $p-value$ of t-test display in Table 1. As can be seen from Table 1, the performance of the two-stage model using $threshold$ is better than the two-stage model using $threshold_{min}$ on F_1 over F_1 matrix. For the t-test, the $p-value$ is less than 0.0001. This is considered to be extremely statistically significant. With regarding to the threshold setting methods, the newly developed threshold setting method significantly outperforms than the threshold setting method developed in [11].

Table 1. $threshold$ vs $threshold_{min}$

	$threshold_{min}$	$threshold$	$p-value$
$F_{\beta=1}$	0.4535	**0.5144**	5.7727E-13

The Different Types of Two-stage Models. Table 2 illustrates the possible combinations of a term-based model integrated with the pattern-based model or a term-based model integrated with another term-based model, where for the efficiency issue, the model that used for the first stage should be a term-based model. In this section, the topic filtering model developed in this study is called

Table 2. Different integrations of filtering models

Integration	Model
Term + Pattern	T-SM (TFM+PTM)
Term + Pattern	BM25+PTM
Term + Pattern	SVM+PTM
Term + Term	TFM+BM25
Term + Term	TFM+SVM

Table 3. T-SM vs other types of two-stage models

	$F_{\beta=1}$	*p-value*
T-SM	**0.5144**	
BM25+PTM	0.4749	2.15271E-07
SVM+PTM	0.4783	2.02454E-06
TFM+BM25	0.4564	4.90958E-08
TFM+SVM	0.4612	4.23648E-07

a Topic Filtering Model(TFM). This term-based model will be integrated with other term-based model to form a term-based + term-based two stage models. The results shown in Table 3 are the comparisons of T-SM with other possible two-stage models. The BM25 and SVM based term weight methods are used in the first stage for the BM25 + PTM and SVM + PTM models. The results show that T-SM significantly outperforms all other two-stage models. The purpose of the topic filtering stage is to move the "noisy" and prepare more "clean" data for pattern-mining stage. As can be seen from the above results TFM can achieve this goal because the design objectives of TFM are different from the traditional filtering models. The TFM has an excellent performance for determining non-relevant information comparing with the traditional models that focus on the performance for determining relevant information.

6 Discussion

As was mentioned previously, the pattern taxonomy mining is sensitive to the data noise. To deal with this phenomenon, a two-stage theory was put forward. In the proposed two-stage model, the rough threshold model were used to remove the majority of the irrelevant documents in the first stage. The goal of the first stage is to produce a relatively small set of mostly relevant documents as the input for the second stage, pattern taxonomy mining.

In theoretical perspective, any incoming document with a larger probability value than the minimum score of positive documents in the training set should be considered as possibly relevant. However, in real life, user profiles can be very uncertain. Using the minimum positive score as the threshold will allow too many irrelevant documents into the second stage. In certain cases, when the user profiles are most specific, using the minimum score as the threshold would

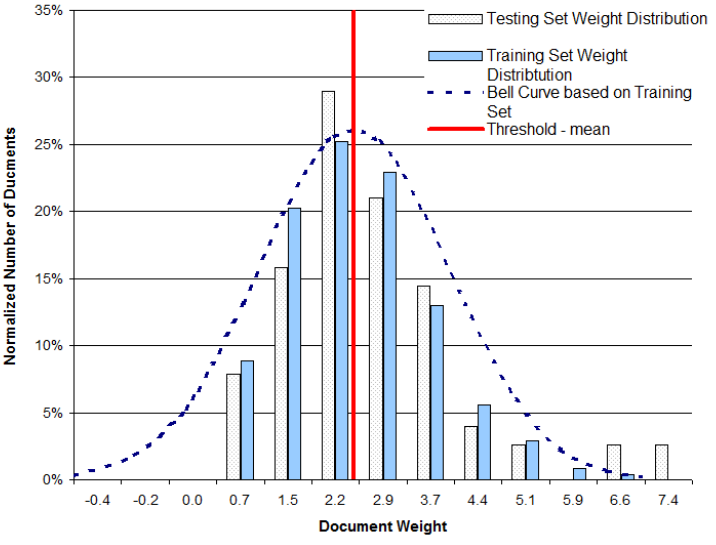


Fig. 1. Threshold by training set mean weight

be appropriate. However, in most cases, the user profiles are not well defined; therefore, the higher score should be used.

Observe the typical document score distributions shown in Figure 1, where the distributions of the positive documents for *Topic* 155 in both the training set and testing set are illustrated. It is clearly illustrated that the use of minimum score of positive documents in the training set can only filter out a few testing documents. In the rough threshold model, the mean score of the positive documents in the training set was used as the threshold instead of the minimum score because it describes the common feature of the positive documents in the training set. Also, a significant improvement was achieved using the common feature as the threshold.

We also observed that many of the positive training sets' scores do not strictly follow the normal distributions. Therefore, the mean plus a fraction of the standard deviation and the skew was used for the rough threshold model.

7 Conclusions

Compared with some other possible types of "two-stage" models, the experimental results confirm that the proposed two-stage IF model (T-SM) significantly outperforms the other models. The substantial improvement is mainly due to the rough sets based threshold optimization method applied to the first stage and the "semantic" based patterns mining and matching applied to the second stage. This research work has delivered a very promising methodology for developing effective and efficient filtering systems based on positive relevance feedback.

References

1. Belkin, N.J., Croft, W.B.: Information filtering and information retrieval: two sides of the same coin? Commun. ACM 35(12), 29–38 (1992)
2. Li, Y., Zhou, X., Bruza, P., Xu, Y., Lau, R.Y.: A two-stage text mining model for information filtering. In: CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, California, USA, pp. 1023–1032 (2008)
3. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley, Reading (1999)
4. Robertson, S.E., Soboroff, I.: The trec 2002 filtering track report. In: TREC (2002)
5. Zhou, X., Li, Y., Bruza, P., Wu, S.T., Xu, Y., Lau, R.Y.K.: Using information filtering in web data mining process. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, Washington, DC, USA, pp. 163–169 (2007)
6. Qin, T., Zhang, X.D., Wang, D.S., Liu, T.Y., Lai, W., Li, H.: Ranking with multiple hyperplanes. In: SIGIR, pp. 279–286 (2007)
7. Jindal, N., Liu, B.: Identifying comparative sentences in text documents. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 244–251. ACM, New York (2006)
8. Ahonen, H., Heinonen, O., Klemettinen, M., Verkamo, A.I.: Applying data mining techniques for descriptive phrase extraction in digital document collections. In: Proceedings of the IEEE Forum on Research and Technology Advances in Digital Libraries (ADL 1998), Santa Barbara, CA, USA, pp. 2–11 (1998)
9. Bayardo, J.: Efficiently mining long patterns from databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 85–93 (1998)
10. Wu, S.T., Li, Y., Xu, Y.: Deploying approaches for pattern refinement in text mining. In: Proceedings of the Sixth International Conference on Data Mining, Washington, DC, USA, pp. 1157–1161 (2006)
11. Li, Y., Zhong, N.: Mining ontology for automatically acquiring web user information needs. IEEE Transactions on Knowledge and Data Engineering 18(4), 554–568 (2006)
12. Wu, S.T., Li, Y., Xu, Y., Pham, B., Chen, P.: Automatic pattern-taxonomy extraction for web mining. In: The IEEE/WIC/ACM International Conference on Web Intelligence, China, pp. 242–248 (2004)

Efficiently Retrieving Longest Common Route Patterns of Moving Objects By Summarizing Turning Regions

Guangyan Huang[1], Yanchun Zhang[1], Jing He[1], and Zhiming Ding[2]

[1] Centre for Applied Informatics, School of Engineering & Science, Victoria University, Australia
abysshuang@gmail.com, {yanchun.zhang,jing.he}@vu.edu.au
[2] Institute of Software Chinese Academy of Sciences
zhiming@iscas.ac.cn

Abstract. The popularity of online location services provides opportunities to discover useful knowledge from trajectories of moving objects. This paper addresses the problem of mining longest common route (LCR) patterns. As a trajectory of a moving object is generally represented by a sequence of discrete locations sampled with an interval, the different trajectory instances along the same route may be denoted by different sequences of points (location, timestamp). Thus, the most challenging task in the mining process is to abstract trajectories by the right points. We propose a novel mining algorithm for LCR patterns based on turning regions (LCRTurning), which discovers a sequence of turning regions to abstract a trajectory and then maps the problem into the traditional problem of mining longest common subsequences (LCS). Effectiveness of LCRTurning algorithm is validated by an experimental study based on various sizes of simulated moving objects datasets.

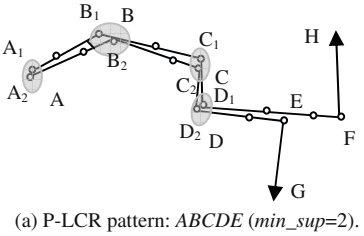
Keywords: spatial temporal data mining, trajectories of moving objects, longest common route patterns.

1 Introduction

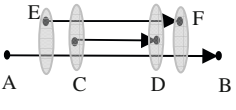
The popularity of online location services provides opportunities to discover useful knowledge from trajectories of moving objects for various applications. This paper addresses the problem of mining longest common route (LCR) patterns. Mining LCR patterns is one of the fundamental issues in mining trajectories of moving objects. LCR patterns are the longest LSPs (Long, Sharable Patterns)[1]. However, different from [1], the aim of this paper is to tackle the following two challenges to retrieve LCR patterns: (1) Accuracy Challenge: As a trajectory of a moving object is generally represented by a sequence of discrete locations sampled with an interval, the different trajectory instances along the same route may be denoted by different sequences of points (location, timestamp) [2] [3] [4]. (2) Efficiency Challenge: the sampling interval can be very small to ensure the accuracy of trajectories but this approach generates a large number of useless points on trajectories [3] [5] [2].

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 375–386, 2011.

To tackle the above two challenges, we propose a novel mining algorithm for LCR patterns based on popular Turning regions (LCRTurning) and explain its effectiveness using examples in Fig. 1. We classify LCR patterns into two classes: Polygon line based LCR (P-LCR) and Direct line based LCR (D-LCR), as shown in Fig. 1 (a) and (b), respectively. We observe that the turning points, where objects change their directions, are always critical and thus we abstract each trajectory by using a sequence of turning points. In Fig. 1 (a), two trajectories are simplified as $A_1B_1C_1D_1FH$ and $A_2B_2C_2D_2EG$, by using turning points. Then, we group all the turning points on both simplified trajectories into different clusters based on their spatial proximity; examples are cluster 1: $\{A_1, A_2\}$, cluster 2: $\{B_1, B_2\}$, cluster 3: $\{C_1, C_2\}$ and cluster 4: $\{D_1, D_2\}$. We define a popular turning region be an area that encloses at least *min_sup* (minimal number of supports) points in a cluster and assume $min\_sup = 2$; examples include Region A, B, C and D. Finally, we unify the two trajectories as two strings: $ABCDFH$ and $ABCDEG$. By the above three steps, problem of mining LCR from trajectories is mapped into the traditional problem of mining Longest Common subsequences (LCS) from strings. Obviously, the common string is $ABCD$ in this example. Then we refine $ABCD$ by $ABCDE$ and $ABCDE$ is a P-LCS pattern in this case. Meanwhile, D-LCR patterns are discovered by another method. For example, the moving object MO1 travels from A to B in Fig. 1 (b), where MO1 also passes other four regions: E, C, D and F, but these four points are not recorded in the trajectory of MO1. So, we retrieve common direct line segments to find D-LCR patterns (e.g. EF in Fig. 1 (b)). Therefore, the LCRTurning algorithm simplifies trajectories by turning points to remove a large number of useless points and then discovers popular turning regions to simplify trajectories further; these tackle Efficiency Challenge. Moreover, it tackles Accuracy Challenge by unifying simplified trajectories based on popular turning regions.



(a) P-LCR pattern: *ABCDE* (*min_sup*=2).



(b) D-LCR pattern: *EF* (*min_sup*=2) and *CD* (*min_sup*=3).

Fig. 1. Two classes of LCR patterns

Our LCRTurning algorithm is different from already exist algorithms and can retrieve LCR efficiently and accurately and we explain as follows. In [2], Douglas-Peucker (DP) algorithm [6] is used to simplify the trajectories, which tackles Efficiency Challenge. We not only validate that DP algorithm is effective to find turning points in single trajectory in our proposed algorithm, but also consider global popularity of a turning region to simplify trajectories further. Moreover, we discover LCR patterns mainly through clustering turning

points other than mainly clustering direct line segments in [2]. In [3], CuTS (Convoy Discovery using Trajectory Simplification) algorithm is provided, which firstly retrieves coarse candidate results based on DP-simplified trajectories and then validate candidates by clustering locations at every timestamp. The coarse searching screens a large volume of points to be checked further to increase the efficiency, but it did not evaluate the accuracy of CuTS since the correct patterns were unknown. In this paper, to evaluate both false positive rates and false negative rates of our LCRTurning algorithm, we develop a benchmark method to denote a trajectory by using the intersections and every trajectory along the same route can be detected exactly by the sequences of intersections. Note that our LCRTurning algorithm do not depend on any road network information. In [5], popular regions visited frequently by moving objects, are used to dynamically define region of interests (RoIs) and then help discover trajectory patterns. However, the popular regions cannot be discovered precisely because the large number of meaningless locations may lead to find false popular regions. Also, if we use intersections as RoIs and define a neighborhood of each intersection to help remove useless non-intersection locations on sampled original trajectories to tackle Accuracy Challenge, it is time consuming to retrieve common patterns since there are a huge number of intersections; our experimental results validate this. The efficiency of our LCRTuring algorithm is validated by an experimental study based on various sizes of moving objects datasets.

The rest of this paper is organized as follows. We model the problem in Section 2. The LCRTurning algorithm is provided in Section 3, while the performance of the proposed algorithm is evaluated in Section 4. Finally, Section 5 concludes the paper.

2 Problem Definition

We formally define basic terms and then model the problem.

Definition 1. *A trajectory of moving object is a sequence of points* $S =< p_1, ..., p_u, ..., p_k >$, *where* $p_u = (x_u, y_u, t_u)$, t_u $(u = 0..k)$ *is a timestamp for a snapshot,* $\forall_{0 \le u < k}$, $t_u < t_{u+1}$, *and* (x_u, y_u) *are 2-D locations.*

Definition 2. *Distance,* $d_{lp}(p, AB)$, *between one point* $p = (x, y)$ *and one direct line segment AB is the minimal distance between p and any point on AB.*

Definition 3. *Turning point is a point* $p_i = (x_i, y_i, t_i)$, *on a trajectory* $S =< p_1, ..., p_u, ..., p_k >$ *that satisfies one of the two conditions: (1) i=1 or k for the start or end points of the trajectory; (2)* $d_{lp}(p_i, p_b p_a) > \lambda$ *(*λ *is a distance threshold), where* $2 \le b, i, a \le k - 1$, *and* p_b *is sampled before* p_i *and* p_a *is sampled after* p_i*; any other point,* p_j $(b < j < a$ *), a point on original trajectory between* p_b *and* p_a *satisfies* $d_{lp}(p_j, p_b p_a) < \lambda$, *where* $d_{lp}(p_j, p_b p_a)$ *is the distance between* p_j *and* $p_b p_a$.

Definition 4. *A turning point simplified trajectory is a sequence of turning points* $TP =< p_1, ...p_i, ...p_w >$, *where* $p_i = (x_i, y_i, t_i)$ *is the* i^{th} *turning point.*

Definition 5. *Interest region. Given the following condition:(1) A is a 2-D region area with arbitrary shape; (2) a set of turning point simplified trajectories* $\Sigma = \{TP_1, ...TP_u, ...TP_m\}$ *in a period of* $[T1, T2]$*, where* $T1$ *and* $T2$ *are timestamps, and a set of points* $F = \{Q_1, ..., Q_i, ...Q_c\}$*, where* $Q_i = (x_i, y_i, t_i) \in TP_1 \cup ...TP_u ... \cup TP_m$ *and* $(x_i, y_i) \in A$*; and (3) min_sup is the minimal number of supports. There are two types of Interest region: (1) If* $c \geq min\_sup$*, then A is a* ***(Explicit) Popular turning region****. (2) If* $c < min\_sup$ *and a set of line segments on different trajectories:* $\{Seg_1, ..., Seg_i, ..., Seg_b\}$ $(i \in [1, b])$*, satisfies two conditions: (1)* $\exists j \forall i (d_{lp}(loc_j, Seg_i) < \lambda)$*, where* $loc_j = (x_j, y_j)$*,* $Q_j = (loc_j, t_j) \in F$ *and* $loc_j \in A$*; (2)* $b \geq min\_sup - c$*, then A is an* ***Implicit popular turning region****.*

Definition 6. *A Longest Common Route (LCR) pattern of moving objects is a route* $\zeta =< r_1, ..., r_i, ..., r_n >$ $(n \geq 2$ *and* r_i $(1 \leq i \leq n$ *) is an interest region) that is visited by a set of moving objects* $MOset = \{MO_1, MO_2, ..., MO_k\}$ *and an LCR pattern satisfies three conditions: (1)* $k \geq min\_sup$ *; (2) k is the maximum value, no other moving objects* $MO_i \notin MOset$ *visit the whole route* ζ*; (3)* ζ *is the longest route visited by the whole set MOset.*

Definition 7. *A Polygon line based LCR (P-LCR) pattern is an LCR pattern which includes at least two popular turning regions.*

Definition 8. *A Direct line based LCR (D-LCR) pattern is an LCR pattern which includes at most one popular turning region and at least one implicit popular turning region.*

Definition 9. *Given N sequences, Longest Common Subsequence (LCS) is the longest subsequence with at least min_sup supports,* $2 \leq min\_sup \leq N$ *.*

Definition 10. *Distance,* d_{ll} *, between two direct line segments AB and CD is the minimal distance between any point p on AB and CD, that is,* d_{ll}*=min* $(d_{pl}(p, CD))$*, where* $p \in AB$ *and* d_{pl} *is defined in Definition 2.*

Definition 2 and Definition 10 are based on the similar definitions in [3]. Then, the problem of mining LCR patterns is modeled as follows. We first detect *turning points* (Definition 3) by the DP algorithm and a single trajectory is denoted by a *turning point simplified trajectory* (Definition 4). We then group all the *turning points* in a set of trajectories into clusters and *turning points* in the same cluster are taken as in the same *popular turning region* (Definition 5). Thus, a trajectory can be abstracted using a sequence of *popular turning region* IDs. Note that we only mark region IDs on the *turning points* but keep the original locations in order to compute the distance between region IDs. Therefore, we can discover *P-LCR patterns* (Definition 7) by mining *LCS* (Definition 9). Finally, we discover *implicit popular turning regions* (Definition 5). Based on d_{lp} (Definition 2), we compute d_{ll} (Definition 10) of two direct line segments. Two direct line segments can be grouped into the same cluster, if d_{ll} is in a bounded error and the angle between the two direct line segments is also in another bounded error. So, for each direct line segment cluster, we discover the *implicit popular turning regions*

through analyzing the overlap part of the direct line segments. This is the process we retrieve *D-LCR patterns* (Definition 8). We prove that *P-LCR* and *D-LCR patterns* compose the whole set of *LCR patterns* (Definition 6).

Theorem 1. *Given a* P-LCR pattern *set* S_{PLCR}*, a* D-LCR pattern *set* S_{DLCR} *and a LCR pattern set* S_{LCR} *for the same input trajectories, then they satisfy: (1)* $S_{PLCR} \cap S_{DLCR} = \Phi$*; (2)* $S_{LCR} = S_{PLCR} \cup S_{DLCR}$*.*

Proof. (1) We use a pair of interest region IDs, P, to denote for a *D-LCR pattern* and a sequence (no less than two) of interest region IDs , H, to denote for a *P-LCR pattern*. $P \in S_{DLCR}$ and $H \in S_{PLCR}$. We firstly may insert the implicit popular turning regions in P into H, denoted by H', if P is included in a piece of segment of H and any of P's ID is not in H yet. There are two cases:

Case 1: a *D-LCR pattern* is fully included in a *P-LCR pattern*, that is $H' \supseteq P$ (this means P is a subsequence of H'), since a *P-LCR pattern* includes no less than two popular turning region IDs and a *D-LCR pattern* includes at most one popular turning region IDs. If $H' \neq P$, then P is not the longest common route and thus P is not a LCR; this also means $P \notin S_{DLCR}$. Therefore, $H' = P$, a *D-LCR pattern* is a sub sequence of a *P-LCR pattern*, in this case, we neglect this *D-LCR pattern*.

Case 2: two lists of moving objects, $MOSet1$ and $MOSet2$ supports H' and P, respectively and a part of a *D-LCR pattern* is included in a *P-LCR pattern*, that is $H' \cap P \subset P$ or $(P - H') \cap H' = \Phi$. If $MOSet1$=$MOSet2$, then this produces a paradox: moving objects that supports both H' and P not only changed direction but also went straight without direction change. Therefore, $MOSet1 \neq MOSet2$ and $H' \cap P = \Phi$. This proves Theorem 1(1).

(2) According to definition 7 and 8, $S_{PLCR} \subseteq S_{LCR}$ and $S_{DLCR} \subseteq S_{LCR}$. Thus, $S_{PLCR} \cup S_{DLCR} \subseteq S_{LCR}$. Suppose except P-LCR patterns and D-LCR patterns, there is another type of patterns, say X pattern, which belong to LCR pattern. That's $S_{PLCR} \cup S_{DLCR} \cup S_X = S_{LCR}$ and $S_X \cap (S_{PLCR} \cup S_{DLCR}) = \Phi$. According to definition 7, a P-LCR pattern includes m popular turning regions, $m \geq 2$. According to definition 8, a D-LCR pattern includes n popular turning regions, $0 \leq n < 2$. Suppose an X pattern includes k popular turning regions, because $S_X \cap (S_{PLCR} \cup S_{DLCR}) = \Phi$, $0 \leq k < 2$ and ($k < 0$ or $k \geq 2$). Thus, $k < 0$. This is not correct, since $k \geq 0$. So, $S_X = \phi$. This proves Theorem 1 (2).

3 Mining Algorithm for Longest Common Route Patterns Based on Turning Regions

In this section, we present the LCRTurning algorithm.

3.1 Discovering Turning Regions

According to Definition 3, DP algorithm [6] [7] is suitable to detect turning points for trajectory simplification. The main idea of the DP algorithm in [7] is given as follows. The first point and the last point of one trajectory are chosen as the

anchor point and float point respectively. For intermediate points, the cut point is the point with the maximum perpendicular distance, which is greater than a pre-defined threshold, λ, to the line connecting anchor and float points. The cut point becomes the new float point for the first segment and the anchor point for the second segment. If no cut point exists, the procedure stops. This procedure is recursively repeated for both segments.

Then we develop a following Clustering Turning Points (CTP) algorithm based on DBSCAN algorithm [8, 9] to group turning points (Definition 3) into popular turning regions (Definition 5):

Step 1. Build neighbour lists of each point on trajectories. The neighbours of point $q(x,y,t)$ must satisfy the criteria that the locations of neighbours are in the neighbourhood circle area with (x,y) as the centre and *eps* as the radius.
Step 2. Build a core point set, I. The number of a core point's neighbors is greater than $MinPts$.
Step 3. All core points are marked "unused". For each unused core point p, put p and p's neighbors into cluster $class\_id$ and mark the point as "used". Any core point r in cluster $class\_id$ will recruit r's neighbors into cluster $class\_id$.
Step 4. Unused points are assigned into a special noise cluster.

Finally, trajectories are denoted by sequences of popular turning region IDs.

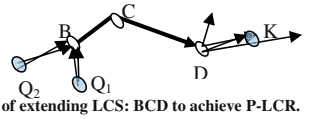
3.2 Retrieving Longest Common Route Patterns

(1) Mining Polygon Line based LCR (P-LCR) Patterns
In the MPLCR algorithm (Algorithm 1), $SuffixTree\_CS$ procedure retrieves the common sequences based on suffix trees [10, 11] and $Remove\_SubSeq$ procedure removes short patterns. Fig. 2 (b) shows an example LCS: BCD. Assuming

Algorithm 1. Mining P-LCR patterns (MPLCR) Algorithm.

Input: A sequences of popular turning region IDs: S_{PTR}; Output: P-LCR patterns.
1. *STr* ← Build suffix tree for S_{PTR};
2. S_{CS} ← **SuffixTree_CS**(*STr*);
3. S_{LCS} ← **Remove_SubSeq** (S_{CS});
4. **For** each $lcs \in S_{LCS}$
5. **For** *LineSet* at f^{th} end /*two ends of a LCS */
6. { **If** $|LineSet| \geq min\_sup$ {
6a. *LineClusterSet* ← **CL_DBSCAN** (*LineSet*, *Angle_eps*, *CL_MinPts*);
6b. **For** each cluster *i* in *LineClusterSet* /* *LineClusterSet*={$L_1, L_2, \ldots, L_m$} */
{ **If** $|L_i| \geq min\_sup$
{$ExtendSet_f$ ← A new interest region, *K*, discovered at the end of the i^{th} (*i*=*min_sup*) longest direct line segment in L_i ;}}} /* Fig 2. (b)*/
7. *PLCRSet* ← Extend *LCS* according to $ExtendSet_1$ and $ExtendSet_2$;
8. Output (*PLCRSet*); }

(a) Algorithm.



(b) Example of extending LCS: BCD to achieve P-LCR.

Fig. 2. Algorithm of Mining P-LCR Patterns

Algorithm 2. Mining D-LCR patterns (MDLCR) Algorithm.

Input: A set of direct line segments with support less than *min_sup*;
Output: D-LCR patterns.

1. *Cluster_DLS*←**DL_DBSCAN**(*DLSet, DL_angle, DL_eps, MinPts*);

2. **For** *each cluster i*

2a. *CutSet* ←**DIP**(*cluster i*); /* Algorithm 3 */

2b. Put the longest subsections of *CutSet* with support no less than *min_sup* into *DLCRSet*;

2c. Output (*DLCRSet*);

Fig. 3. Algorithm of Mining D-LCR Patterns

BCD has a support list of $m \geq min\_sup$ (min_sup =3) moving objects, we may check the direct line segments that connect to the two ends of the LCS, shown in Fig. 2(b). Note that only the trajectories of moving objects that support BCD are checked. We develop CL_DBSCAN (CL denotes Constraint Line) based on the DBSCAN algorithm to cluster the direct line segments at each end of an LCS by checking whether they have the same moving direction. Then, the i^{th} ($i = min\_sup$) longest direct line segment is the newly discovered part of the route; DK in Fig. 2(b) shows an example. Given $min\_sup = 2$, Q_1BCDK and Q_2BCDK in Fig. 2(b) are two P-LCR patterns.

(2) Mining D-LCR Patterns

In the MDLCR algorithm (Algorithm 2), DL_DBSCAN (DL denotes Direct Line) clusters direct line segments based on two criteria to determine the similarity of two direct line segments: (1) Spatial closeness to each other; (2) Similar direction. We use d_{ll} (Definition 10) to define the spatial closeness and measure the direction similarity by computing the angle by $f(u, v) = arccos(\frac{u \bullet v}{|u| * |v|})$, $f \in [0, 180°]$, where u and v are two vectors, and $|u|$ and $|v|$ denote the lengths of the two vectors respectively. DL_angle and DL_eps are the threshold angle and the threshold distance. The DIP Algorithm (Algorithm 3) discover implicit popular turning regions (Definition 5). Suppose DL_DBSCAN generates n clusters denoted by Φ_1 , Φ_2, ..., Φ_n and $\Phi = \Phi_1 \cup \Phi_2 \cup .....\Phi_n \cup \Phi_{noise}$. Firstly, average direction, $\overrightarrow{v}$, of each direct line segment cluster is computed, then we rotate the axes so that the X axis is parallel to $\overrightarrow{v}$. The most related work that compute a representative trajectory in a direct line segment cluster is given in [12]. Taking a different approach, we rotate a cluster of direct line segments to help discover cuts (or implicit popular turning regions). Given $RDLSCluster = \{AB, CD, FE\}$, $A(x1, y1)$, $B(x2, y2)$, $C(x3, y3)$, $D(x4, y4)$, $E(x5, y5)$ and $F(x6, y6)$ in $X - Y$ coordinate, we can compute $A'(x1', y1')$, $B'(x2', y2')$, $C'(x3', y3')$, $D'(x4', y4')$, $E'(x5', y5')$ and $F'(x6', y6')$ in $X' - Y'$ coordinate by Eq. 1.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix}. \quad (1)$$

Given $x6' < x1' < x3' < x5' < x4' < x2'$ in Fig. 4 (b), for each direct line segment cluster, we find all the cut points and sort them. The list of sorted cuts is $< x6', x1', x3', x5', x4', x2' >$, so there are 5 cut sections: $[x6', x1']$, $[x1', x3']$,

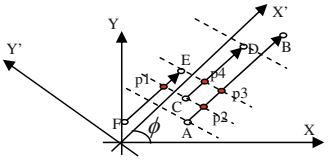
$[x3', x5']$, $[x5', x4']$ and $[x4', x2']$. For each cut section $[xi', xj']$ included in any line segments in $RDLSCluster$, the number of supports are counted. Note that one direct line segment here may be supported by more than one moving objects. Finally, all the longest consecutive cut sections with supports no less than min_sup are put into $DLCRSet$. In Fig. 4 (b), given min_sup=3, the original direct line section according to cut section $[C(x3'), E(x5')]$ is an D-LCR pattern. Short D-LCR patterns may be neglected.

Algorithm 3. Discover Implicit Popular turning regions (DIP) Algorithm.

Input: A cluster of direct line segments; **Output**: implicit popular turning regions;

1. Put average direction angle of *Cluster_DLS[i]* into *avAngle*;
2. **For** each direct line segment *j* in *Cluster_DLS[i]* {
   2a. *RDLSCluster[j]*←**Rotate**(*avAngle*, *Cluster_DLS[i][j]*); \\ see Eq. (1)
   2b. *CutSet* ← *RDLSCluster[j].x1'*; *CutSet* ← *RDLSCluster[j].x2'*;}
3. **Qsort**(*CutSet*); Remove redundant one from *CutSet*;
4. **For** each point *j* in *CutSet*
5. **For** each direct line segment *k* in *Cluster_DLS[i]*
6. **If** (*[CutSet[j],CutSet[j+1]*∈ *[RDLSCluster[k].x1', RDLSCluster[k].x2'])*
   {*CutSupport[j]+=Cluster_DLS[i][j].nSupport*;}
7. Put the longest subsections of *CutSet* with support (>*min_sup*) into *DLCRSet*;

(a) Algorithm.



(b) Cut points in a cluster of direct line segments.

Fig. 4. Discover Implicit Popular turning regions (DIP) Algorithm

4 Performance Evaluations

In our experiments, we used a Network-based Generator of Moving Objects [13] to generate various sizes of moving object datasets described in Table 1, where we chose an open road map of Oldenburg (a city in Germany) as the network input. Several parameters of our algorithms are given in Table 2. We set $DL\_eps = eps$ and $MinPts = 2$. We now briefly describe our experimental setup. Given the trajectories exactly denoted by intersections provided by the simulator, we ran MPLCR algorithm to retrieve a set of longest common sub-sequences of intersections supported by at least min_sup moving objects as benchmarks (shown in Table 1), which were used to measure the accuracy of our proposed methods. We studied three cases of proposed algorithms: DP2, DP5 and DP10, where DPi denote using DP ($\lambda = i$) to abstract trajectories as input for LCRTurning algorithm (e.g. CTP, MPLCR and MDLCR). First, we studied the optimal eps and optimal DL_angle by setting $min\_sup = 2$. Then, we observed the scalability based on datasets with various data sizes shown in Table 1. We adopted the same optimal values of eps and DL_angle in the scalability test. We use false positive rate and false negative rate to measure the accuracy of patterns.

Table 1. Moving Objects Datasets Using Generator in [13]

Parameter	Test 1	Test 2	Test 3	Test 4	Test 5
Max. Timestamps	60	120	180	240	300
Number of MOs	1,295	1,600	1,900	2,200	2,500
Number of Points	65,012	121,833	179,314	230,082	264,348
Data size (Mb)	3.2	6.2	9.2	11.7	13.8
Total pattern length Benchmarks(Meter)	24470	54509	81502	103613	117839

Table 2. Parameters of LCRTurning Algorithm

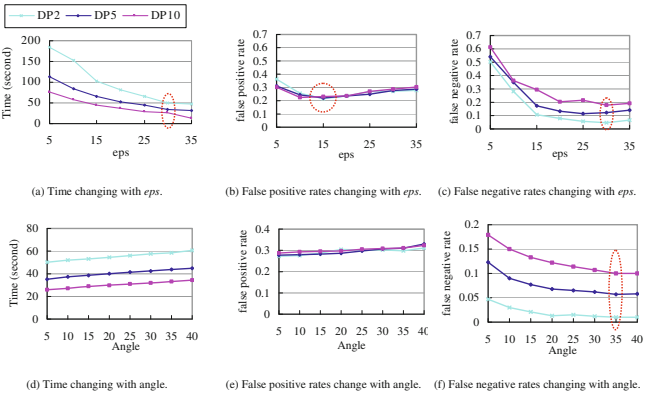
Parameter	Explanation	Parameter	Explanation
Angle_eps	Threshold angle in *CL_DBSCAN*	λ	Pre-defined distance threshold in *DP*
DL_Angle	Threshold angle in *DL_DBSCAN*	*eps*	Threshold distance in *DBSCAN*
DL_eps	Threshold of d_{ll} in *DL_DBSCAN*	*min_sup*	Threshold number of supports for a pattern.
MinPts	In three clustering algorithms: *DBSCAN*, *CL_DBSCAN* and *DL_DBSCAN*.		

Given the benchmark total pattern length, L', and the total retrieved pattern length, $L = L_c + L_e$, where L_c is the correct retrieved pattern length and L_e is the error retrieved pattern length. False positive rate is L_e/L' and false negative rate is $(L' - L_c)/L'$. All experiments were conducted on an IBM Laptop with Microsoft Windows XP, Genuine Intel 1.83GHz CPU and 512MB main memory. We implemented the proposed algorithms mainly by using C++, and the suffix tree based retrieving LCS algorithm was implemented by Java.

4.1 Optimal *eps* and *DL_angle*

Letting $L\_Angle = 5^o$, $L\_eps = eps$ and min_sup=2, we ran our proposed algorithms on dataset Test1. Time spent on retrieving LCR patterns is plotted in Fig. 5 (a); it shows that the greater the value of *eps* is, the less time spent on running algorithms DP2, DP5 and DP10. DP10 performed more efficiently than both DP2 and DP5, and DP5 was more efficient than DP2. We analyze the accuracy of patterns shown in Fig. 5(b)(c). The whole trend of the three algorithms in Fig. 5 (b) is that the lowest values of false positive rate are all at *eps*=15. Also, DP2 performs better than DP10 and DP5, with the lowest false negative rate (4.7%) at *eps*=30 as shown in Fig. 5 (c). Overall, the best effectiveness and false negative rate is achieved at *eps*=30, while the best false positive rate is achieved at *eps*=15. Since all three algorithms performed more efficiently at *eps*=30 than at *eps*=15, *eps*=30 was the optimal one with best effectiveness.

Letting $L\_eps = eps = 30$, we studied the optimal DL_angle. We plotted time changing with angles in Fig. 5(d), where DP2 consumes more time than DP5 and DP10. DP2 achieves best false negative rate, near to zero at $L\_Angle = 35$ in Fig. 5 (f). Fig. 5 (f) also shows that the greater the angle is, the lower the value of false negative rate, while Fig. 5 (e) shows that the value of false positive rate is increased with the angle. Overall, DP2 is more accurate but less efficient. The best false positive rate and the best time efficiency are achieved at $L\_Angle = 5$ and the best false negative rate is achieved at $L\_Angle = 35$.



(a) Time changing with *eps*. (b) False positive rates changing with *eps*. (c) False negative rates changing with *eps*.

(d) Time changing with angle. (e) False positive rates change with angle. (f) False negative rates changing with angle.

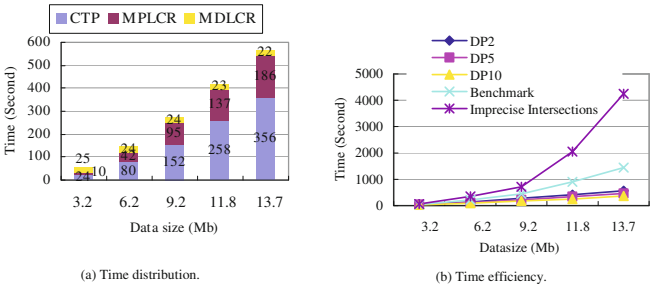
Fig. 5. Analysis of optimal parameters

4.2 Efficiency and Accuracy

Based on the results in Section 4.1, we let *eps*=30 and $L\_Angle = 35^o$ to evaluate the efficiency and accuracy of our algorithms in various dataset sizes.

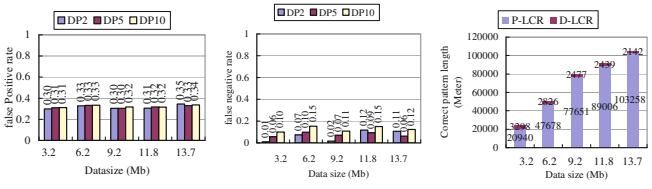
Our proposed algorithms are more efficient than both Benchmark and Imprecise Intersections methods as shown in Fig. 6 (b). In the Imprecise Intersections method, where the intersections provided by the simulator are not repeated exactly, we use the CTP algorithm with $eps = 0.1$ to group imprecise intersections into region clusters and mine LCS of region IDs. The Imprecise Intersections method is not efficient, since a huge number of imprecise intersections make the CTP algorithm time-consuming. The time costs of DP2, DP5 and DP10 are all nearly linear, while the time cost of Benchmark and Imprecise Intersections climbed quickly with increased data size, up to around four times and twelve times respectively that spent on DP10. The Imprecise Intersections method - that has worst time efficiency - also validates that using turning regions to retrieve LCR patterns is efficient. Fig. 6 (a) shows in DP2, the time spent on CTP is increased around 14 times from 24s at data size of 3.2Mb to 356s at 13.7Mb, and the time spent on MPLCR is increased around 18 times from 10s to 186s, while time spent on MDLCR is unchanged at around 24s. The time spent on CTP is around twice that spent on MPLCR.

The accuracy of our proposed algorithms is shown in Fig. 7(a)(b). When data size is increased, false positive rate stays in the range of 30%-35%, and false negative rate is lower, always less than 15%. Fig. 7(c) shows that the correct lengths of D-LCR patterns are nearly the same in various data sizes, which also only constitute a very small part of the total lengths. The correct length of P-LCR patterns is the major part: from 86% at data size=3.2Mb to 98% at data size =13.7Mb. Although we evaluate our proposed algorithm by using simulating traffic moving objects, LCRTurning algorithm can be used widely to any trajectories of moving objects that may not follow a network, since turning regions are discovered automatically from trajectories.



(a) Time distribution. (b) Time efficiency.

Fig. 6. Time Efficiency



(a) False Positive Rate. (b) False Negative Rate. (c) P-LCR vs. D-LCR.

Fig. 7. Accuracy

5 Conclusions

We proposed a novel LCRTurning algorithm to retrieve longest common route patterns from moving object trajectories. The LCRTurning algorithm have wide applications for any trajectories that may not follow a network. Experimental results validated that the LCRTurning algorithm is more efficient than the algorithms that retrieve LCR patterns by using intersections while achieving reasonable accuracy.

Acknowledgement

This work was partially supported by Australian Research Council projects (Grant No. LP100200682, LP0882957, LP100100554), as well as the National Natural Science Foundation of China (Grant No. 70602034, 70621001, 60970030) and CAS/SAFEA International Partnership Program.

References

1. Gidofalvi, G., Pedersen, T.B.: Mining Long, Sharable Patterns in Trajectories of Moving Objects. GeoInformatica 13(1), 27–55 (2009)
2. Cao, H., Mamoulis, N., Cheung, D.W.: Mining frequent spatio-temporal sequential patterns. In: ICDM 2005, pp. 82–89 (2005)

3. Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S., Shen, H.: Discovery of convoys in trajectory databases. In: VLDB, pp. 1068–1080 (2008)
4. Gudmundsson, J., Kreveld, M.V., Speckmann, B.: Efficient detection of patterns in 2D trajectories of moving points. Geoinformatica 11, 195–215 (2007)
5. Giannotti, F., Nanni, M., Pedreschi, D., Pinelli, F.: Trajectory pattern mining. In: SIGKDD 2007, pp. 330–339 (2007)
6. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer 10(2), 112–122 (1973)
7. White, E.R.: Assessment of line generalization algorithms using characteristic points. The American Cartographer 12, 17–27 (1985)
8. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: SIGKDD 1996, pp. 226–231 (1996)
9. He, J., Huang, G., Zhang, Y., Shi, Y.: Cluster analysis and optimization in color-based clustering for image abstract. In: ICDM Workshops, pp. 213–218 (2007)
10. McCreight, E.M.: HA space-economical suffix tree construction algorithmH. Journal of the ACM 23(2), 262–272 (1976)
11. Dan, G.: Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology. Cambridge University Press, USA (1997)
12. Lee, J.-G., Han, J., Whang, K.-Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD 2007, pp. 593–604 (2007)
13. Brinkhoff, T.: A framework for generating network-based moving objects. GeoInformatica 6(2) (2002)

Automatic Assignment of Item Weights for Pattern Mining on Data Streams

Yun Sing Koh[1], Russel Pears[2], and Gillian Dobbie[1]

[1] Department of Computer Science, University of Auckland, New Zealand
{ykoh,gill}@cs.auckland.ac.nz
[2] School of Computing and Mathematical Sciences, AUT University, New Zealand
rpears@aut.ac.nz

Abstract. Research in Weighted Association Rule Mining (WARM) has largely concentrated on mining traditional static transactional datasets. Whilst there have been a few attempts at researching WARM in a data stream environment, none have addressed the problem of assigning and adapting weights in the presence of concept drift, which often occurs in a data stream environment. In this research we experiment with two methods of adapting weights; firstly, a simplistic method that recomputes the entire set of weights at fixed intervals, and secondly a method that relies on a distance function that assesses the extent of change in the stream and only updates those items that have had significant change in their patterns of interaction. We show that the latter method is able to maintain good accuracy whilst being several times faster than the former.

Keywords: Weighted Items, Valency Model, Data Stream Mining.

1 Introduction

Ever since its inception, data stream mining has remained one of the more challenging problems within the data mining discipline. Although extensively researched, many unsolved problems remain. Yet streams are an increasingly important and rich source of data that can yield valuable knowledge when mined effectively. Data from a wide variety of application areas ranging from online retail applications such as online auctions and online bookstores, telecommunications call data, credit card transactions, sensor data and climate data are but a few examples of applications that generate vast quantities of data on a continuous basis. Furthermore, data produced by such applications are highly volatile with new patterns and trends emerging on a continuous basis.

Association Rule Mining (ARM) is one approach to mining such data streams. However a well known limitation of ARM is that it has the potential to generating a large number of rules, most of which are trivial or of no interest to the decision maker. One method of overcoming this problem is to weight items in terms of importance so that rules that only contain high weight items are presented to the user. The crucial factor in ensuring the success of this approach is the

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 387–398, 2011.

assignment of weights to items. The typical approach is for users to supply the weights through a subjective process based on their specialized knowledge of the domain involved. While this subjective weight assignment may be feasible in an environment where data is stable, we believe that it will not be effective in the context of data stream mining where data is subject to constant change. Such change or concept drift is bound to invalidate any weight assignment over a period of time. In this research we thus propose an algorithm to adapt item weights to reflect changes that take place in the data stream.

Recent research by Koh et al. [1] has shown that automatic inference of weights from a transaction graph is an effective method of ranking items according to their interest measure, or Valency. The Valency model assigns a weight to an item depending on how strongly it interacts with other items. However the Valency model was not designed to operate in a dynamic environment and in this research we focus on extending the Valency model to adapt weights by capturing the changing patterns of interaction over time.

The rest of the paper is organized as follows. In the next section we review work that has been done on item weighting and briefly cover work done on incremental methods employed in a data stream environment. In Section 3 we describe the Valency model in greater detail, while in Section 4 we describe our methodology for incremental update of weights. Our experimental results are presented in Section 5. The paper concludes in Section 6 where we assess to what extent we have achieved our goal of evolving weights over a stream as well as presenting our thoughts for future research in this area.

2 Background and Related Work

There has been much work in the area of pattern mining in data streams [2,3,4]. However, there has been very little research specifically directed at mining weighted patterns in a data stream environment. The few attempts to address this problem have not employed automatic methods for item weight assignment and maintenance.

Ahmed et al. [5] proposed a sliding window based technique WFPMDS (Weighted Frequent Pattern Mining over Data Streams) which employs a single pass through the data stream. They utilized an FP tree to keep track of the weighted support of items and hence their mining scheme was essentially based on the FP tree algorithm with the major difference being that weighted support was used in place of raw support. Kim et al. [6] proposed a weighted mining scheme based on two user defined thresholds $t1$ and $t2$ to divide items into infrequent, latent and frequent categories. Items with *weighted support* $< t1$ were categorized into the infrequent category and those with *weighted support* $> t2$ were grouped into the frequent category; all others were taken to be latent. Items in the infrequent category were pruned while items in the other two categories were retained in a tree structure based on an extended version of the FP tree. However in common with Ahmed et al., their weight assignment scheme was both subjective and static, and as such would degrade in the face of concept

drift that occurs in many real world situations. When concept drift occurs such mining schemes essentially take on the character of frequent pattern mining as items are not re-ranked in terms of importance, and changes in their weighted support only reflect changes taking place in support, rather than item rank.

The main challenge to be overcome in weighted association rule mining over a stream is to be able to adapt item weights according to the changes that take place in the environment. Such changes force adjustments to previously assigned weights in order to make them better reflect the new data distribution patterns. We argue that it is not possible for domain experts to play the role that they normally do in static data environments. Firstly, due to the open ended nature of a data stream, it will be very difficult for a human to constantly keep updating the weights on a regular basis. Secondly, even if they regularly invest the time to keep the weights up to date, it will be very hard, if not impossible to keep track of changes in the stream for applications such as click stream mining, fraud detection, telecommunications and online bookstore applications, where the number of items is very large and the degree of volatility in the data can be high. Our weight assignment scheme is based on the Valency model due to its success in generating high quality rules as reported in [1]. However, as connectivity between items plays a critical role in the Valency model, it is necessary to monitor the stream and identify which items have had significant changes in their interactions with other items in order to efficiently adapt the weights. In a stream containing N items, a naive method of performing such an estimation would be to check the interactions between all pairs of items which has a worst case time complexity of $O(N^2)$. With N having values in the tens of thousands or even hundreds of thousands, such an approach would be extremely inefficient or even prohibitive in the case of high speed data streams. Thus the challenge boils down to finding an efficient and accurate estimation method that has a worst case time complexity closer to the ideal value of $O(N)$. We describe such a method in Section 4.

3 Valency Model

The Valency model, as proposed by Koh et al. is based on the intuitive notion that an item should be weighted based on the strength of its connections to other items as well as the number of items that it is connected with. Two items are said to be connected if they have occurred together in at least one transaction. Items that appear often together relative to their individual support have a high degree of connectivity and are thus weighted higher. The total connectivity c_k of item k which is linked to n items in its neighborhood is defined as:

$$c_k = \sum_{i}^{n} \frac{count(ki)}{count(k)} \tag{1}$$

The higher the connectivity c_k of a given item k, the higher its weight should be, and vice versa. While high connectivity is a necessary condition for a high weight,

it is not considered to be sufficient by itself, as the weighting scheme would then be too dependent on item support which would bias weighting too much towards the classical (un-weighted) association rule mining approach. With this in mind, a purity measure was defined that encapsulated the degree to which an item could be said to be distinctive. The smaller the number of items that a given item interacted with, the higher the purity and vice versa. The reasoning here is that an item should not be allowed to acquire a high weight unless it also has high purity, regardless of its connectivity. The role of purity was thus to ensure that only items with high discriminative power could be assigned a high weight, thus reducing the chances of cross support patterns from manifesting.

Formally, the purity of a given item k was defined as:

$$p_k = 1 - \frac{\log_2(|I_k|) + \log_2(|I_k|)^2}{\log_2(|U|)^3} \tag{2}$$

where $|U|$ represents the number of unique items in the dataset and $|I_k|$ represents the number of unique items which are co-occurring with item k. Purity as defined in Equation 2 ensures that the maximum purity of 1 is obtained when the number of items linked with the given item is 1, whereas the purity converges to the minimum value of 0 as the number of linkages increases and becomes close to the number of items in the universal set of items. The logarithmic terms ensure that the purity decreases sharply with the number of linkages in a non linear fashion.

The Valency contained by an item k, denoted by v_k as the combination of both the purity and the connectivity components is defined below:

$$v_k = \beta.p_k + (1 - \beta).\sum_{i}^{n} \frac{count(ki)}{count(k)}.p_i \tag{3}$$

where β is a parameter that measures the relative contribution of the item k over the items that it is connected with in the dataset and is

$$\beta = \frac{1}{n}\sum_{i}^{n} \frac{count(ik)}{count(k)}$$

We use the Valency contained by an item as its weight.

4 Our Weight Adaptation Methodology

We use a sliding window mechanism to keep track of the current state of the stream. Our weight adaptation scheme uses a distance function to assess the extent of change in the stream. In order to keep overheads within reasonable bounds the distance computation is only applied periodically, or after every b number of instances (i.e. a block of data of size b) are processed. Thus a block consists of a number of overlapping windows.

We now present a 2-phased approach to capturing interactions between items and detecting change points in the data stream. In phase 1 an initial block of

data is read and item neighborhoods are built in the form of 1-level trees in order for an initial assignment of weights to be made. In the second phase, weights are adapted through the use of a distance function as subsequent blocks of data arrive in the data stream.

In phase 1 (Read in initial block): Each transaction is read and items are sorted in their arrival order in the stream. Whenever a new item is detected, a 1-level tree is created with that item as its root. Figure 1 shows the trees created for a transaction containing 4 items, A B, C, and D. The 1-level trees so formed enable easy computation of the connectivity and purity values for each item. Once the purity and connectivity is calculated for an item its weight is then computed by applying Equation 3.

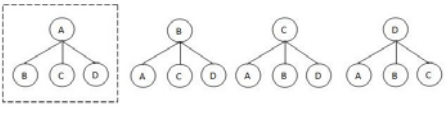


Fig. 1. Graph Node

In phase 2 (Read in subsequent blocks): The oldest transaction in the window is deleted from the inverted index matrix and the new transaction is inserted. At the conclusion of the current block a distance function is applied to each item to determine whether or not its joint support with neighboring items (i.e. items that it occurs with) has changed significantly enough, in which case it is recomputed from a buffer containing transactions in the current block. The update to the joint support of an item A with another item B triggers an update to the connectivities of both items, A and B. The purity of an item is only updated if a new item is introduced or if an item disappears from the current block. We recalculate weights for any items that have had their connectivity or purity values changed.

4.1 Data Structure: Inverted Index Matrix

We use an inverted index to buffer transactions in the current block. Transactions are represented by a prefix tree structure. The index key consists of the item id. The support of the item is also stored with the index key in order to speed up retrieval. Each index value consists of a triple $< i, j, k >$ where i is a pointer to the next key, j is the tree prefix that the item occurs in, and k is the support of the item with respect to tree j. Given an initial transaction (A, B, C, D, E), arriving in this order, Table 1(a) shows the initial entry, whereby the initial tree id is 1. Given the arrival of another transaction (B, A, C, F), we now sort this transaction into the order: (A, B, C, F). Here the entry for $A(2, 1, 1)$ is updated to (2,1,2); this is repeated for each subsequent entry until item C where an entry (6, 1, 1) is added to slot 2 of the index value array to point to its successor F. A new slot is required for entry (6,1,1) as this represents a new branch in the prefix tree structure. Table 1(b) shows the updated index after the arrival of the second transaction.

Table 1. Inserting into the modified Inverted Matrix

(a) Before Insertion of transaction (B, A, C, F)

loc	Index	Transactional Array	
		1	2
1	(A,1)	(2,1,1)	
2	(B,1)	(3,1,1)	
3	(C,1)	(4,1,1)	
4	(D,1)	(5,1,1)	
5	(E,1)	(ϕ,1,1)	

(b) After Insertion

loc	Index	Transactional Array	
		1	2
1	(A,2)	(2,1,2)	
2	(B,2)	(3,1,2)	
3	(C,2)	(4,1,1)	(6,1,1)
4	(D,1)	(5,1,1)	
5	(E,1)	(ϕ,1,1)	
6	(F,1)	(ϕ,1,1)	

Table 2. Deleting from the modified Inverted Matrix

(a) Before Deletion

loc	Index	Transactional Array	
		1	2
1	(A,4)	(2,1,3)	(4,2,1)
2	(B,4)	(3,1,3)	(6,3,1)
3	(C,3)	(4,1,2)	(6,1,1)
4	(D,2)	(5,1,1)	(5,2,1)
5	(E,2)	(ϕ,1,1)	(ϕ,2,1)
6	(F,2)	(ϕ,1,1)	(ϕ,3,1)

(b) After Deletion

loc	Index	Transactional Array	
		1	2
1	(A,3)	(2,1,2)	(4,2,1)
2	(B,3)	(3,1,2)	(6,3,1)
3	(C,2)	(4,1,2)	
4	(D,2)	(5,1,1)	(5,2,1)
5	(E,2)	(ϕ,1,1)	(ϕ,2,1)
6	(F,1)	(ϕ,3,1)	

The inverted matrix structure supports efficient deletion of transactions as well. Table 2 shows the state of the index after the arrival of a number of additional transactions. Given the oldest transaction (A, B, C, F) in the current window, with tree id 1, we find the position of A with tree id 1 and reduce the count corresponding to this position by one. Here the entry for $A(2, 1, 2)$ is updated to (2,1,1); this is repeated for each subsequent entry until item F is encountered. As the count for the F(ϕ,1,1) is updated to (ϕ,1,0), this entry is removed from the matrix.

The inverted matrix structure is used to buffer transactions across only two blocks, since our distance and estimation functions only require the comparison of the state of the current block with that of the previous block. Another advantage of using this data structure is the efficiency in finding the joint support between two items. Given two items, A and C, the joint support AC, is obtained by carrying out an intersection operation between the items based on their tree ids. For each tree that this pair occurs in we extract the minimum of their tree count values and accumulate this minimum across all trees that this pair participates in. As A and C appear in only one tree with tree id 1, the minimum count across tree 1 is taken, which happens to be 2.

The inverted matrix while being efficient at computing the joint support of item pairs does not efficiently support the maintenance of an item's connectivity. To update the connectivity of an item, its neighborhood needs to be enumerated. This would require excessive traversal of the inverted matrix. Hence we also maintain one level trees for each item, as described earlier. For every new item which appears in the stream, an entry is added to the inverted matrix and a link is established to its one level tree where that item appears as a root node.

4.2 Distance Function

To calculate the weights for an item using the Valency model mentioned in Section 3, we track the joint support of an item with other items in its neighborhood. In order to prevent expensive and unnecessary updates, we use a distance function $d(X, Y)$ to assess the extent of change in joint support between a given item X with another item Y. Let $S_n(X)$ be the actual support of item X in block n and $\hat{S}(X)_n$ be its support conditional on no change taking place in the connectivity between it and item Y in consecutive windows $n-1$ and n. The distance function is then given by:

$$d(X, Y) = |\hat{S}(X)_n - \mathrm{S}(X)_n|$$

where $\hat{S}(X)_n$ is given by:

$$\hat{S}(X)_n = \mathrm{S}(XY)_{n-1} \frac{\mathrm{S}(Y)_n}{\mathrm{S}(X)_n} \frac{\mathrm{S}(X)_{n-1}}{\mathrm{S}(Y)_{n-1}} \frac{1}{\mathrm{C}(X, Y)_{n-1}}$$

where n is the current block number.

Rationale. Under the assumption of no pattern drift in the connectivity between X and Y we have:

$$\mathrm{C}(X, Y)_n = \mathrm{C}(X, Y)_{n-1}$$

where $\mathrm{C}(X, Y)_{n-1}$, $\mathrm{C}(X, Y)_n$ are the connectivities between X and Y in blocks n-1 and n respectively. We estimate:

$$\hat{S}(X)_n = \frac{\hat{S}(XY)_n}{\mathrm{C}(X, Y)_n} = \frac{\hat{S}(XY)_n}{\mathrm{C}(X, Y)_{n-1}}$$

In the absence of concept drift the joint support between X and Y will remain stable only if the factor

$$\frac{S(Y)_n}{S(Y)_{n-1}} \frac{S(X)_{n-1}}{S(X)_n}$$

remains stable between blocks. Our intention is to trap any significant changes to the join support arising out of significant changes to this factor. Under the assumption of stability in this factor and no concept drift between X and Y we have:

$$\hat{S}(XY)_n = S(XY)_{n-1} \frac{S(Y)_n}{S(Y)_{n-1}} \frac{S(X)_{n-1}}{S(X)_n}$$

and so we reformulate $\hat{S}(X)_n$ as:

$$\hat{S}(X)_n = \mathrm{S}(XY)_{n-1} \frac{\mathrm{S}(Y)_n}{\mathrm{S}(X)_n} \frac{\mathrm{S}(X)_{n-1}}{\mathrm{S}(Y)_{n-1}} \frac{1}{\mathrm{C}(X, Y)_{n-1}}$$

We now have an estimation of the support of X that is sensitive to changes in the stream. Any significant departure from our assumption of no drift will now

be trapped by the distance function $d(X,Y) = |\hat{S}(X)_n - \mathrm{S}(X)_n|$ since it includes the drift sensitive term $\hat{S}(X)_n$.

In order to assess which values of $d(X,Y)$ are significant we note that the function is a difference between the sample means of two random variables which are drawn from unknown distributions. In order to assess significance between the means we make use of the Hoeffding bound [7]. The Hoeffding bound is attractive as it is independent of the probability distribution generating the observations. The Hoeffding bound states that, with probability $1-\delta$, the true mean of a random variable, r, is at least $\overline{r} - \epsilon$ when the mean is estimated over t samples, where

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2t}}$$

R is the range of r. In our case the variable r is denoted by $\hat{S}(X)_n - \mathrm{S}_n(X)$, which has a range value R of 1, and the number of samples $t = b$, the block size. Thus for any value of $d(X,Y) > \epsilon$ our assumption of no concept drift is violated and an update of the join support $S_n(XY)$ is required from the transactions in our buffer for block n.

However we were also conscious of the fact that even small changes in the joint support $S_n(XY)$ not signalled by the distance function can accumulate over all items Y in item X's neighborhood. Although each of the deviations are small individually they could become significant when added over all the links between X and its neighbors. We thus decided to add a correction factor (but not an update) to the joint support even in the event that $d(X,Y) \leq \epsilon$. Our experimentation showed the importance of adding this correction factor, with it in place the precision of identifying high weight items increased quite significantly.

The correction factor is given by:

$$S(XY)_n = \mathrm{S}(XY)_{n-1} + \mathrm{S}(XY)_{n-1} * \frac{(\mathrm{S}(X)_n - \hat{S}(X)_n) * \mathrm{C}(X,Y)_{n-1}}{2}$$

The correction factor basically adds or subtracts, as the case may be, a factor to the joint support that is equal to the product of the connectivity with the median value of the deviation between $\mathrm{S}(X)_n$ and $\hat{S}(X)_n$. We carry out a complete update of the joint support of every pair of items after p number of blocks. This is to ensure that the estimation remains within a reasonable range. The parameter p has to be set at a reasonable value to achieve a good balance between precision and efficiency. In all our experimentation we set p to 10.

5 Evaluation

We report on a comparative analysis of our incremental approach, referred to as *WeightIncrementer*, with the simple approach of updating all item weights periodically at each block of data, referred to as *Recompute*. Our experimentation used both synthetic and real world datasets. We compared both approaches

on execution time. In addition, we tracked the precision of WeightIncrementer relative to Recompute. Using Recompute we tracked the identity of items having weights in the top k% and used the identity of these items to establish the precision of WeightIncrementer. The precision is given by: $\frac{|WI \bigcap R|}{|R|}$ where WI is the set of items returned by WeightIncrementer as its top $k\%$ of items; R is the set of items returned by Recompute as its top $k\%$ of items.

Our experimentation on the synthetic data was conducted with the IBM dataset generator proposed by Agrawal and Srikant [8]. To create a dataset D, our synthetic data generation program takes the following parameters: number of transactions $|D|$, average size of transactions $|T|$, average size of large itemsets $|I|$, and number of large itemsets $|L|$. We chose not to compare our methods with other existing item weight schemes for data streams, as they require explicit user defined weights, thus making any comparison inappropriate.

5.1 Precision

In this set of experiments, we used the following parameters $|T|20|I|4|D|500K$ and the number of unique item as 1000. Figure 2 shows the variation of Precision with the number of large $|L|$ items which were varied in the range from 5 to 25, in increments of 5. The top k% parameter was varied from 10% to 60%. The overall precision of WeightIncrementer vis-a-vis Recompute across all such experiments (i.e. over all possible combinations of L and k parameters) was 0.9. To measure the effect of the ϵ parameter in the Hoeffding bound we varied the reliability parameter δ. We used values of 0.00001, 0.001, 0.01, 0.1, and 0.2 for δ and tracked the precision for the top 20% of the items using a block size of 50K. Table 3 displays the results of precision based on the varying values. Overall there is a general trend where Precision increases with δ (i.e. decreasing ϵ).

Table 3. Variation of Precision with δ

Large Itemset	δ value				
	0.2	0.1	0.01	0.001	0.00001
5	0.95	0.95	0.95	0.95	0.90
10	0.81	0.81	0.80	0.80	0.78
15	0.91	0.91	0.80	0.80	0.80
20	0.94	0.94	0.94	0.94	0.89
25	0.97	0.89	0.88	0.88	0.88

5.2 Execution Time

We next compared the two methods on execution time. Both algorithms used the same data structure as described in the previous section. In this set of experiments, we used the following parameters $|T|20|I|4|L|20$ and the number of unique items as 1000. We varied the number of transactions ($|D|$) from 500K to 5M, with a block size of 100K. Figure 3 shows the execution time for the two methods for varying values of δ.

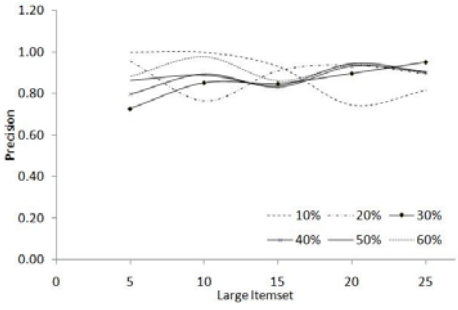


Fig. 2. Precision based on datasets L5 to L25

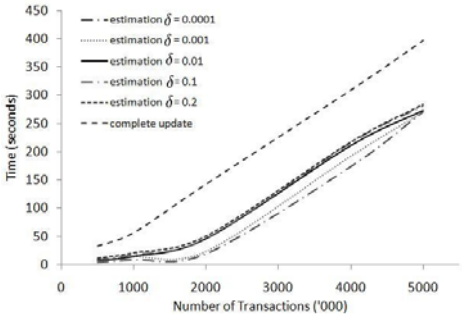


Fig. 3. Execution Time

In the experiments above, with a δ value of 0.00001 the percentage of updates to joint support (considering all possible pairs of items that interact with each other in a given block) was 27% whereas when the δ value was increased to 0.2 the percentage of updates increased to 37%. We also tracked the speedup achieved by WeightIncrementer with respect to Recompute. The speedup, defined as the ratio of the speed of WeightIncrementer to the speed of Recompute, ranged from a minimum of 1.4 to a maximum of 8.3.

5.3 Evaluating Drift

To evaluate the performance of WeightIncrementer in capturing drift we modified the IBM data generator to inject known drift patterns to track the sensitivity of WeightIncrementer to drift detection. We select x random points in time which:

- Introduce large itemsets: The main motivation is to simulate emerging patterns. Given the original set of large itemsets introduced, we hold off introducing a particular large itemset until we reach a predetermined point. Once we reach that point we introduce a large itemset into the stream with a probability of occurrence that increases incrementally over time.
- Remove large itemsets: The main motivation is to simulate disappearing patterns. When we reach a predetermined point, we remove a large itemset from the stream by incrementally decreasing its probability of occurrence over a period of time.

To evaluate whether our approach captures drift, we compare the results of the two methods, WeightIncrementer, with Recompute. We track the success rate of Recompute in detecting the large itemsets in the top 50% that were deliberately injected/removed and use this as a baseline to measure the performance of WeightIncrementer. We denote the ratio of the success rate of WeightIncrementer to Recompute as the *Hit Ratio*. If WeightIncrementer is able to capture all the items that were deliberately injected/removed relative to Recompute, then the hit ratio would be 1. In this experiment we modified the $|T|20|I|4|L|40|D|500K$

dataset; we use a block size of 50K and a δ value of 0.1. We injected various drift points into the dataset. Table 4 displays the results for the various drift points.

Table 4 shows that WeightIncrementer was able to detect both emerging as well as disappearing patterns with a minimum hit ratio of 74%, demonstrating that it can effectively adapt item weights and re-rank items with a high degree of precision in the presence of concept drift.

Table 4. Drift Analysis based on Hit Ratio

Drift Points	Large Items Injected	Large Items Removed	Hit Ratio
10	5	5	0.88
10	10	0	0.74
10	0	10	1.00
20	10	10	0.95
20	20	0	0.80
20	0	20	0.95

5.4 Real World Dataset: Accident

To evaluate our algorithm on a real dataset, we choose to evalute our algorithm on the accident dataset [9,10]. The accident dataset was provided by Karolien Geurts and contains (anonymized) traffic accident data. In total, 340,184 traffic accident records are included in the data set. In total, 572 different attribute values are represented in the dataset. On average, 45 attributes are filled out for each accident in the data set.

We ran experiments using multiple block size at 25K, 40K, and 60K. We then looked at the precision of the items based on the top 50%. We also compared the speedup of WeightIncrementer against Recompute. As Table 5 shows, the speedup ranged from 1.5 to 1.7. From Table 5 we note that the precision based on the top 50% of items was around the 87% mark. We also observed that the precision was not sensitive to changes in the δ value, remaining fairly constant across the δ range.

Table 5. Results based on the Accident Dataset

Block Size	WeightIncrementer			Recompute
	Precision (Top 50%)	Percentage of Updates	Time (s)	Time (s)
25K	0.86	0.39	776	1187
40K	0.87	0.36	736	1377
60K	0.90	0.38	575	955

Table 6. Hit Ratio vs δ (Block 60K)

Delta	0.2	0.1	0.01	0.00001
Precision	0.90	0.90	0.89	0.88

Overall we notice that the results from the accident dataset is consistent with that of synthetic data. The distance function works well to detect drift and estimate joint support. However we do acknowledge that under certain situations the run time performance of WeightIncrementer approaches that of Recompute; this happens when there are rapid changes, or fluctuations in the data stream. This will cause a substantial drift amongst all items in the dataset. When this

occurs, most of the joint support values need to be updated. However this only occurs in unusual circumstances with certain types of datasets, and we would not normally expect to see this kind of trend in a typical retail dataset displaying seasonal variations.

6 Conclusion

In this paper we proposed an algorithm to adaptively vary the weights of items in the presence of drift in a data stream. Item weights are assigned to items on the basis of the Valency model, and we formulated a novel scheme for detecting and adapting the weights to drift. Our approach reduces the number of updates required substantially and increases efficiency without compromising on the quality of the weights. We tested our drift detection mechanism on both synthetic and real datasets. Our evaluation criteria focussed on precision and efficiency in runtime. We were able to achieve Precision rates ranging from 86% to the 95% mark whilst achieving substantial speedup in runtime for both synthetic and real-world data.

References

1. Koh, Y.S., Pears, R., Yeap, W.: Valency based weighted association rule mining. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010. LNCS, vol. 6118, pp. 274–285. Springer, Heidelberg (2010)
2. Chang, J.H., Lee, W.S.: Finding recent frequent itemsets adaptively over online data streams. In: KDD 2003, pp. 487–492. ACM, New York (2003)
3. Chi, Y., Wang, H., Yu, P., Muntz, R.: Moment: maintaining closed frequent itemsets over a stream sliding window. In: ICDM 2004, pp. 59 –66 (November 2004)
4. Leung, C.S., Khan, Q.: Dstree: A tree structure for the mining of frequent sets from data streams. In: ICDM 2006, pp. 928 –932 (December 2006)
5. Ahmed, C.F., Tanbeer, S.K., Jeong, B.S.: Efficient mining of weighted frequent patterns over data streams. In: 10th IEEE International Conference on High Performance Computing and Communications, pp. 400–406 (2009)
6. Kim, Y., Kim, W., Kim, U.: Mining frequent itemsets with normalized weight in continuous data streams. JIPS 6(1), 79–90 (2010)
7. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association 58(301), 13–30 (1963)
8. IBM Almaden Research Center: Synthetic data generation code for associations and sequential patterns (1997), http://www.almaden.ibm.com/softwarequest
9. Goethals, B.: Fimi dataset repository, http://fimi.cs.helsinki.fi/data/
10. Geurts, K., Wets, G., Brijs, T., Vanhoof, K.: Profiling high frequency accident locations using association rules. Transportation Research Record: Journal of the Transportation Research Board 1840, 123–130 (2003)

Predicting Private Company Exits Using Qualitative Data

Harish S. Bhat[1] and Daniel Zaelit[2]

[1] University of California, Merced, 5200 N. Lake Rd., Merced, CA 95343, USA
hbhat@ucmerced.edu
[2] SVB Analytics, 555 Mission St., San Francisco, CA 94105, USA
dzaelit@svb.com

Abstract. Private companies backed by venture capitalists or private equity funds receive their funding in a series of rounds. Information about when each round occurred and which investors participated in each round has been compiled into different databases. Here we mine one such database to model how the private company will exit the VC/PE space. More specifically, we apply a random forest algorithm to each of nine sectors of private companies. Resampling is used to correct imbalanced class distributions. Our results show that a late-stage investor may be able to leverage purely qualitative knowledge of a company's first three rounds of funding to assess the probability that (1) the company will not go bankrupt and (2) the company will eventually make an exit of some kind (and no longer remain private). For both of these two-class classification problems, our models' out-of-sample success rate is 75% and the area under the ROC curve is 0.83, averaged across all sectors. Finally, we use the random forest classifier to rank the covariates based on how predictive they are. The results indicate that the models could provide both predictive and explanatory power for business decisions.

1 Introduction

Venture capitalists (VC's) face the challenge of choosing a few outstanding investments from a sea of thousands of potential opportunities. A VC funds a startup company with cash in exchange for an equity stake. From this point of view, it may appear that the dynamics of the transaction are similar to that of an investor buying shares of a publicly traded company. Such appearances are false. When a VC funds a startup, the VC often takes an active role in managing the startup, providing expertise and advice in both managerial and technical areas. In this way, the experience and wisdom of the VC's who invest in a startup directly influence the startup's trajectory.

When confronted with a company they have not seen before, one question that potential investors would like to be able to answer is: how will this company eventually *exit* the private equity space? In this study, we assume that the final outcome of a private company will be one of five outcomes. The private company can (1) go bankrupt, (2) proceed via an initial public offering (IPO) to become a

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 399–410, 2011.

publicly traded company, (3) be subject to a leveraged buyout (LBO), (4) merge with or be acquired by another company, or (5) stay private.

Our goal in this paper is to use information about *who* invests in a private company and *when* these investments are made to predict how the company will exit. Our prediction is generated by a statistical model inferred from data available through the Private Equity module of ThomsonONE, a data set formerly known as VentureXpert. Numerous academic libraries have access to this database, and it is often used as a source of data for research papers in the VC/PE space—see [2, 6, 5, 12, 13]. Here we develop a method for converting each VC- or PE-backed company in the database into a list of numerical and nominal attributes with a class label corresponding to one of the five possible states; we then use this list of labeled instances to train and test a classifier.

The random forest algorithm [3], [10, Chap. 15] and other machine learning algorithms such as support vector machines and boosting are available as free codes, implemented in a variety of languages and environments. Such algorithms have proven useful in a wide variety of applications. Though it seems very natural to leverage machine learning algorithms and large databases to model the exits of VC-/PE-backed companies, to the best of our knowledge, this is the first study to do so. As such, this paper represents a first attempt at solving the problem.

Our analysis shows that a late-stage investor may be able to use knowledge of a company's first three rounds of funding together with a random forest classifier to assess the probability that the company will not go bankrupt, and also to assess the probability that the company will eventually make an exit of some kind (and no longer remain private). For both of these two-class classification problems, our models' average success rate across all sectors is 75% and the average area under the ROC curve is 0.83.

In what follows, we discuss the details of our procedure, starting from the data, proceeding to issues of representation, investor ranking and instance resampling, and then on to specific working models and their associated results.

2 Data Extraction and Representation

For this study, we focused on the following attributes:

- The year in which the company was founded. See the right panel of Figure 1 for a histogram of the inception years for all companies in Sector 6 (energy) used in this study—the most populated decade is the decade from 2000 to the present. Other sectors' distributions are similar.
- The company's sector, encoded as a four-digit number.
- The *rounds*, i.e., dates on which the company received funding.
- A list of historical investors in the company. This list includes each investor's name, type, and a list of the rounds in which that investor participated.
- The company's exit status.

Table 1. Summary of exit types by broad sector category for 77,160 private companies. All companies are either formerly or currently VC- or PE-backed.

	Exit					
Sector	**Bankrupt**	**IPO**	**LBO**	**M&A**	**Private**	**Totals**
1xxx: Communications	1540	826	295	2073	3289	8023
2xxx: Computer	3197	1541	653	4872	9941	20204
3xxx: Electronics	511	604	215	853	1852	4035
4xxx: Biotech/Pharma	283	458	67	509	1652	2969
5xxx: Medical/Health	704	740	412	1228	2787	5871
6xxx: Energy	204	287	150	238	850	1729
7xxx: Consumer	1317	865	1715	1395	4997	10289
8xxx: Industrial	614	473	1009	974	2763	5833
9xxx: Other	2374	1675	2636	2307	9215	18207
Totals	10744	7469	7152	14449	37346	77160

All of these attributes have to do with *who* invested in the private company and *when* the investment was made. Notably, the amount of money invested by the investor in each round of funding, as well as the pre- and/or post-money valuations of the companies are absent. In short, our study makes use of qualitative rather than quantitative features of a private company's investment history.

Let us elaborate on a few of the attributes mentioned above. The company's exit status is, in the original data set, a nominal attribute with 12 possible values. Since exit status is the class variable, we group a few of these categories together to reduce the number of classes from 12 to five. We list here the five class labels in italics together with the original exit types contained in each class:

1. *Bankrupt:* Defunct, Bankruptcy - Chap. 7, Bankruptcy - Chap. 11
2. *IPO:* Went Public
3. *LBO:* LBO
4. *M&A:* Acquisition, Merger, Pending Acquisition
5. *Private:* Active, Other, In Registration, and Private Company (Non-PE)

The company's market sector is encoded as a four-digit number. The first digit of this four-digit number gives us a broad sector categorization, as seen in the left-most column of Table 1, which also shows the breakdown of exits by sector. One can readily see two trends. First, the classes are imbalanced, necessitating the use of a resampling procedure described in Section 3.1.

Second, different sectors behave differently:

- For sector 2xxx, only 3.23% of companies had an LBO exit, while for sector 8xxx, 17.30% of companies exited via LBO.
- For sector 6xxx, 16.60% of exits are IPO and 13.77% of exits are M&A. For sector 1xxx, 10.30% of exits are IPO and 25.84% of exits are M&A.

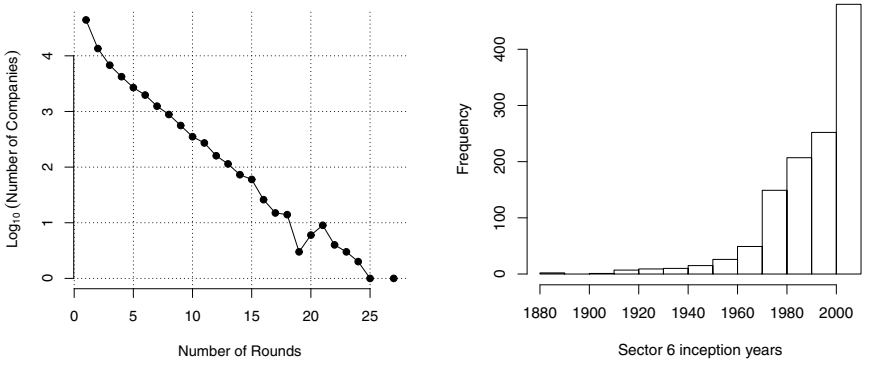


Fig. 1. In the left panel, we plot $\log_{10}$ of the number of companies in the database with x rounds of funding, as x goes from 1 to 27. Linear decay in this plot shows that the same plot with a non-logarithmic y-axis would feature exponential decay. In the right panel, we plot the distribution of inception years for Sector 6 (energy) companies.

It is plausible that the reason the percentages differ so much from one sector to the next is that the factors influencing success/failure differ greatly from one sector to another. For these reasons, in the present study, we shall segregate the data by the broad sectors indicated in Table 1.

2.1 Social Network Ranking

Here we explain how we turn the investor name into an attribute. The social network of coinvestment plays a key role. There are 9545 unique investors in our data set, so we seek a mapping from the set of all investor names to the set of integers $P = \{1, 2, 3, \ldots, 9545\}$.

Let each investor be a node, and join two nodes by an edge if the two investors both invested in the same company at some point of time. To repeat, the coinvestment need not occur at the same time. Once we form the adjacency matrix for this social network, we sort investors by degree. Ties are broken simply by using the order in which we encounter the investor as we parse the data. Once the investors are sorted by degree, we have our mapping: the investor is mapped to its position $p \in P$ in the sorted list.

In the sorted list, the top two investors, Undisclosed Firm and Individuals, are placeholders that do not correspond to any one firm. The next 10 investors are: J. P. Morgan Partners (FKA: Chase Capital Partners), New Enterprise Associates, Inc., Intel Capital, Kleiner Perkins Caufield & Byers, Oak Investment Partners, Sequoia Capital, Goldman, Sachs & Co., Mayfield Fund, HarbourVest Partners LLC, and Bessemer Venture Partners. These names should be familiar to those who follow the VC/PE space, indicating that even a rough social network ranking does correspond to intuitive/anecdotal rankings of VC's and PE funds.

Next we turn to the investor type. This is a nominal attribute with 18 possible values: Development, Buyouts, Seed Stage, Balanced Stage, Recap, Unknown, Energy, Early Stage, Expansion, Fund of Funds, Mezzanine Stage, Later Stage, Turnaround, Distressed Debt, Real Estate, Other Private Equity, Secondary Funds, and Generalist. There is effectively a 19th possible value when the type of the investor is not listed, i.e., the datum is missing. We let Q be the set of 19 possible investor types.

2.2 Mapping Companies to N-tuples

In what follows, we use the term vector to mean an N-tuple $\mathbf{x} = (x_1, x_2, \ldots, x_n)$; one N-tuple represents one company. All the ingredients are in place to define a function that maps companies to vectors. One issue is that the number of rounds of funding enjoyed by a private company varies from one company to the next. Let $n(x)$ be the number of companies that have received precisely x rounds of funding; Figure 1 shows $\log_{10} n(x)$ versus x. The graph is approximately linear, indicating an exponential decay of $n(x)$. There are two considerations to make:

- The maximum number of rounds for any company is 27, yet 92.5% of companies have at most 5 rounds of funding.
- From the point of view of applicability, a model that predicts exit type accurately using *fewer* rounds worth of investor information is preferable.

Based on both considerations, we develop a round-by-round representation of the data. We find that there are a maximum of 31 investors in any round of funding. One round of funding then corresponds to a vector $(\mathbf{p}, \mathbf{q}) \in P^{31} \times Q^{31}$, where P and Q were both defined in Section 2.1. We have $\mathbf{p} = (p_1, p_2, \ldots, p_{31})$, and each p_j is the result of mapping the j-th investor name to P using social network ranking. We also have $\mathbf{q} = (q_1, q_2, \ldots, q_{31})$, and each $q_j \in Q$ is the investor type for investor j.

We see, then, that the representation of a company consists of a number of distinct rounds. We use superscripts to denote the round number. Then, in a model where we retain only the first five rounds of funding, a company C is represented by a vector $C = (\mathbf{h}, \mathbf{p}^1, \mathbf{q}^1, \mathbf{p}^2, \mathbf{q}^2, \mathbf{p}^3, \mathbf{q}^3, \mathbf{p}^4, \mathbf{q}^4, \mathbf{p}^5, \mathbf{q}^5)$.

Besides the information contained in the investor lists, we have a relatively small amount of information that we represent by a vector $\mathbf{h}$. For the model with k rounds of funding, we have $\mathbf{h} \in \mathbb{Z}^{3+k}$, with h_1 equal to the precise four-digit sector code, h_2 equal to the year in which the company was founded, $h_3 = k$, and h_4 through h_{3+k} equal to integer representations of the dates on which the k rounds of funding occurred.

Note that entry-wise addition of two N-tuples generally results in an N-tuple that is not a meaningful representation of any possible company. This lack of linearity excludes a host of statistical methods. This is in contrast to a "bag of words" representation of our data, where we would represent the investor list for one company by a vector $\mathbf{v} \in \mathbb{R}^{9545}$, where v_k represents the number of times that investor k participated in a round of funding for that

company. This representation of the data does have a linear structure and lends itself to models based on matrix factorizations such as the SVD, yielding latent semantic analysis-type models [7]. Though linear models based on the SVD have performed very well on other problems, we found through detailed testing that for our problem, such models suffered from poor predictive power and extremely long computation times. The latter was due to the higher-dimensional spaces incurred by the bag of words representation. For this reason, we moved away from a linear representation of the data to the $(\mathbf{p}, \mathbf{q})$ structure described above.

2.3 Missing Entries

The most striking thing about our representation of the data set is the relatively large number of missing entries incurred. To see how this arises, consider that one company's first round may involve three investors while another company's first round may involve 30. In the first instance, only the first three components of $\mathbf{p}^1$ and $\mathbf{q}^1$ would be populated with meaningful information—the remaining 28 components are missing. In the second instance, there would be only one missing component in each of $\mathbf{p}^1$ and $\mathbf{q}^1$. As long as we wish to retain as much information per round as we have on hand, our representation of the data will lead to missing entries.

Both the missing entries and the lack of vector space structure point to random forests as an appropriate class of models for this problem. Classification/decision tree algorithms upon which random forests are based contain natural methods for estimating missing data. Breiman's tests [3] indicate that random forests can yield accurate models even with 80% missing data.

3 Model Development and Results

In this work, we focus on two two-class problems: distinguishing companies labeled as "bankrupt" from those that are not, and distinguishing companies labeled as "private" from those that are not.

We develop models that make predictions using only the first three funding rounds. We discard all rounds later than round three, and we cap the "number of rounds" entry h_3 of $\mathbf{h}$ so that it is at most equal to three.

3.1 Resampling and Cross-Validation

As can be seen from Table 1, the bankrupt vs. non-bankrupt problem will be highly imbalanced regardless of sector. The imbalance causes problems for all classifiers that we have tried. The problem manifests in a classifier that always predicts "non-bankrupt", yielding an area under the ROC curve close to 0.5, i.e., a perfectly useless model, even if its overall accuracy is anywhere from $70-90\%$. To avoid this issue, for any training set that we feed to the random forest, we sample with replacement from the training set to form a new training set with uniform class distribution. We do not touch the test set. To summarize:

1. Let X = collection of all labeled vectors for one of the two-class problems for one of the sectors.
2. Randomly partition X into K disjoint subsets $\{S_i\}_{i=1}^{K}$.
3. For $i = 1 : K$,
   (a) Let $\mathbf{S} = \bigcup_{j \neq i} S_j$. Sample with replacement from $\mathbf{S}$ to form a new training set $\tilde{\mathbf{S}}$ with uniform class distribution.
   (b) Train a random forest with training set $\tilde{\mathbf{S}}$.
   (c) Test the random forest on S_i.
4. Aggregate the test results from all K folds of cross-validation.

Similar approaches have been discussed by Breiman et al [4]. The imbalance is not as acute but still exists for the private vs. non-private problem, so we employ the resampling procedure for that problem as well.

3.2 Results

All results will be for random forests with 80 trees per forest and 25 randomly chosen attributes per tree. The results are computed using Weka RandomForest [9]. We have found that the test results—both overall correctness and area under the ROC curve—are relatively insensitive to the parameters chosen. For example, varying the number of trees from 35 to 160 in steps of 10 yields ROC areas and overall correctness within 5% of the results quoted below.

In Weka, we have built models using a number of different classifiers appropriate for the attributes and instances in our data set. Even with the resampling procedure described above, the following methods yielded models with poorer predictive power than random forests: logistic regression, support vector machines (with standard kernels), and neural networks. Meta-classifiers such as boosting and bagging performed well and deserve investigation in future work.

Bankrupt vs. Non-Bankrupt Problem. Across all sectors, we find our model performs best for companies in the energy sector (sector 6). The classifier's overall accuracy is 83.4%. The confusion matrix in this case is as follows:

	predicted bankrupt	predicted non-bankrupt
truly bankrupt	167	37
truly non-bankrupt	250	1275

Here the positive class is "bankrupt" and the negative class is "non-bankrupt." Let T/F denote true/false and P/N denote positive/negative. Then we define

$$\text{Positive Precision} = \frac{TP}{TP + FP}, \quad \text{Positive Recall} = \frac{TP}{TP + FN} \tag{1}$$

$$\text{Negative Precision} = \frac{TN}{TN + FN}, \quad \text{Negative Recall} = \frac{TN}{TN + FP} \tag{2}$$

We compute these metrics to assist with decision-making. Each quantity is an estimate of a conditional probability:

$$\text{Pos Precision} = P(\text{truly} + \mid \text{predict} +) \quad \text{Pos Recall} = P(\text{predict} + \mid \text{truly} +)$$
$$\text{Neg Precision} = P(\text{truly -} \mid \text{predict -}) \quad \text{Neg Recall} = P(\text{predict -} \mid \text{truly -})$$

What we notice for Sector 6 is high negative precision, i.e., when the model says that a company is not going to be bankrupt, there is a 97.2% chance it will not go bankrupt. However, when the model says that a company is going to go bankrupt, there is only a 40% chance that it will truly go bankrupt. There are two reasons why this happens:

First, the original data set is rich with examples of non-bankrupt companies, and poor with examples of bankrupt companies. This is purely a function of how the data was gathered—companies that have gone bankrupt already have no incentive to give their historical details to ThomsonONE, and because information on private companies need not be reported publicly, ThomsonONE has no way of finding out about all past bankrupt companies.

Second, given that positive and negative recall are above 0.8, it may well be the case that if our trained models were merely tested on data sets with a much larger number of bankrupt companies, the performance would be much improved. Right now our algorithm predicts bankrupt in over 80% of the cases where the company truly is bankrupt (positive recall = 0.819), but unfortunately, our data set is only 11% bankrupt.

Added together, the two reasons just presented indicate that if the model were trained on a data set that included a more rich set of bankrupt companies, the positive precision would increase.

In addition to the above metrics, there is the ROC curve, formed by accounting for not only the classifier's prediction but also the value of its margin function for each instance—for more details about the construction of ROC curves, see [8]. The curve indicates that a practical decision-making system can be designed based on the margin. When the margin is high, i.e., when we are at the part of the ROC curve near $(0, 0)$, the classifier is consistently correct, giving the curve a large positive slope. This implies that when the margin is high, the classifier gives a useful and trustworthy prediction.

Similar results can be noted across all sectors, as shown in Table 2. ROC curves for all 9 sectors are plotted in the left and right panels of Figure 2. We have separated the ROC curves into two panels merely to enable the reader to distinguish them.

Private vs. Non-Private Problem. Here the model performs much more uniformly across all sectors. This time, let us examine the performance for the largest sector, Sector 2, comprising companies in the general area of computers. The confusion matrix is:

	predicted private	predicted non-private
truly private	2312	767
truly non-private	765	2217

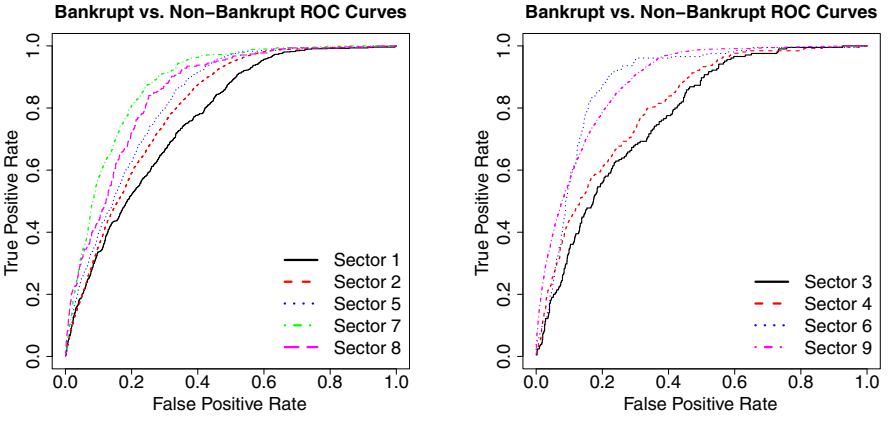


Fig. 2. ROC curves for the Bankrupt vs. Non-Bankrupt classification problem. Each curve corresponds to test set results for a sector-specific random forest model.

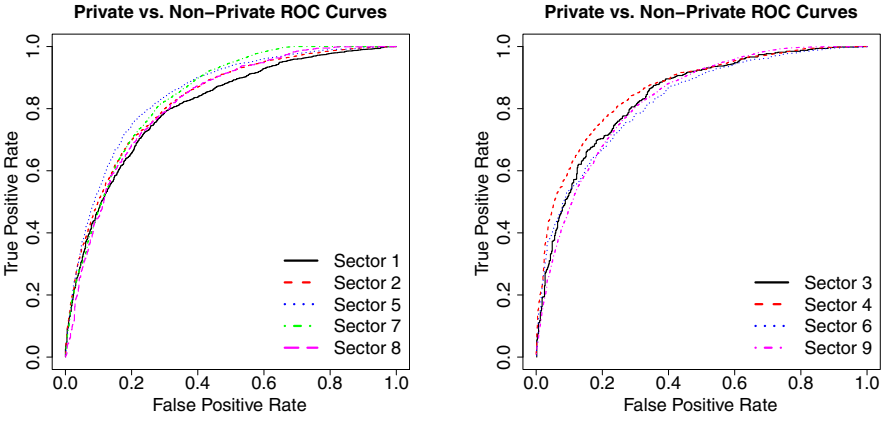


Fig. 3. ROC curves for the Private vs. Non-Private classification problem. Each curve corresponds to test set results for a sector-specific random forest model.

We use the same definitions as given above in (1-2), but now the positive class is "private" and the negative class is "non-private." All four metrics are very close to each other: positive precision = 0.751, positive recall = 0.751, negative precision = 0.743 and negative recall = 0.743. The classifier correctly classifies 74.7% of all instances, and the area under the ROC curve is 0.828.

Very similar results can be noted across all sectors, as shown in Table 3. ROC curves for all 9 sectors are plotted in the left and right panels of Figure 3. Again, we have separated the ROC curves into two panels merely to enable the reader to distinguish them.

Table 2. Sector-by-sector results for the binary classification problem with "Bankrupt" as the positive (+) class and "Non-Bankrupt" as the negative (-) class. "AUC" stands for area under the ROC curve. Results show metric-wise consistency and sector-wise variation. Averaged across all sectors, the AUC is 0.83 and the accuracy is 0.75.

	Random Forest Results					
Sector	**+Precision**	**+Recall**	**-Precision**	**-Recall**	**AUC**	**Accuracy**
1: Communications	0.331	0.742	0.913	0.644	0.765	0.663
2: Computer	0.306	0.816	0.950	0.651	0.801	0.677
3: Electronics	0.264	0.634	0.933	0.743	0.770	0.729
4: Biotech/Pharma	0.241	0.614	0.952	0.797	0.803	0.780
5: Medical/Health	0.277	0.756	0.956	0.731	0.824	0.734
6: Energy	0.400	0.819	0.972	0.836	0.880	0.834
7: Consumer	0.352	0.838	0.970	0.774	0.876	0.782
8: Industrial	0.297	0.726	0.961	0.798	0.850	0.790
9: Other	0.348	0.835	0.969	0.765	0.881	0.774

Table 3. Sector-by-sector results for the binary classification problem with "Private" as the positive (+) class and "Non-Private" as the negative (-) class. AUC stands for area under the ROC curve. Results are consistent across both metrics and sectors. Averaged across all sectors, the AUC is 0.83 and the accuracy is 0.75.

	Random Forest Results					
Sector	**+Precision**	**+Recall**	**-Precision**	**-Recall**	**AUC**	**Accuracy**
1: Communications	0.812	0.726	0.657	0.758	0.809	0.739
2: Computer	0.751	0.751	0.743	0.743	0.828	0.747
3: Electronics	0.793	0.719	0.702	0.779	0.838	0.746
4: Biotech/Pharma	0.774	0.721	0.789	0.832	0.860	0.783
5: Medical/Health	0.795	0.771	0.755	0.780	0.847	0.776
6: Energy	0.760	0.696	0.711	0.773	0.825	0.734
7: Consumer	0.755	0.804	0.777	0.723	0.840	0.765
8: Industrial	0.758	0.764	0.735	0.729	0.821	0.747
9: Other	0.747	0.743	0.750	0.754	0.828	0.748

Ranking the Covariates. As detailed by Breiman [3], random forests provide estimates of variable importance. In Weka, the built-in RandomForest module does not include this feature; we have utilized an extension of the module developed by Livingston [11]. In the table below, we rank our attributes (or covariates) by their importance in the random forest. The importance is given as a RawScore averaged across 10 rounds of cross-validation. For reasons of space, we include in Table 4 only the top 10 attributes for Sector 6: results for other sectors show the same general grouping of attributes.

Table 4. Ranking of attributes for sector 6 random forest models, with bankrupt vs. non-bankrupt results on the left and private vs. non-private results on the right.

RawScore	Attribute	RawScore	Attribute
99.9	Date of rnd 1	89.1	Date of rnd 1
58.1	Type of inv 1 in rnd 1	52.5	Type of inv 1 in rnd 1
32.0	Identity of inv 1 in rnd 1	34.4	Four-digit sector code
27.5	Four-digit sector code	29.2	Identity of inv 1 in rnd 1
17.4	Total # of rnds	22.4	Inception year
17.2	Inception year	15.0	Type of inv 2 in rnd 1
13.1	Type of inv 2 in rnd 1	13.3	Total # of rnds
12.4	Type of inv 1 in rnd 2	12.1	Type of inv 1 in rnd 2
10.2	Date of rnd 2	10.3	Date of rnd 2
6.76	Type of inv 1 in rnd 3	6.79	Type of inv 3 in rnd 1

There are several clear trends to discern from the ranking. Early rounds of funding matter more than later rounds of funding. The type of an investor matters just as much if not more than its identity. Finally, the rankings for both two-class problems show remarkable similarity, both in terms of the order of the ranking and the clustering of the RawScore values in certain intervals. The top four most important attributes are the same for both two-class problems.

4 Discussion and Conclusion

Having performed this study, we see three main ideas for improving the model using currently available data.

First, from Table 4, we see that the identity of the first investor in round one is one of the most important attributes for the random forest models built in this paper. Since this identity consists of the investor's social network ranking, we are left to believe that a more informative social network may lead to better predictions of company exits. The network used in this study ignores temporal details such as the fact that investor A may be completely divested from a startup company by the time that investor B decides to invest. In this case, our network prescribes a connection between the two investors that is not present in reality. Another point is that we have formed *one* network for all investors/companies; forming different networks for each sector may yield better models.

Second, based on our knowledge of the data set, when we view the rankings in Table 4, we infer that attributes that have very low percentages of missing entries (such as the dates of the rounds of funding, which are *never* missing) are much more important for the model's predictive power. We therefore believe that a better understanding of missing entries may yield a more predictive model. Indeed, there may be something significant to learn from (a) the number of investors in each round and (b) which investors do and do not participate in a given round of funding. This is analogous to wisdom from the winners of the Netflix prize, who found that modeling which movies *were* and *were not* rated by a user improved their predictions [1].

Finally, as hinted above, we have only begun to explore other ensemble classifiers such as boosting and bagging. It is likely that combining random forests with other models will yield a model that beats our current results—the only question is how to search for this combination in a principled fashion.

We form two main conclusions: (1) applying resampling and random forests to qualitative data in the VC/PE-space does indeed yield models with useful predictive and explanatory power; and (2) a late-stage investor who has purely qualitative knowledge of a company's first three rounds of funding can use this information to improve his/her understanding of that company's future trajectory. Overall, the results indicate that data mining can be used to provide both predictive and explanatory power for VC decisions.

References

1. Bell, R.M., Koren, Y.: Lessons from the Netflix prize challenge. SIGKDD Explorations 9(2), 75–79 (2007)
2. Bottazzi, L., Rin, M.D., Hellmann, T.: Who are the active investors? Evidence from venture capital. Journal of Financial Economics 89, 488–512 (2008)
3. Breiman, L.: Random Forests. Machine Learning 45(1), 5–32 (2001)
4. Chen, C., Liaw, A., Breiman, L.: Using Random Forest to Learn Imbalanced Data. Tech. Rep. 666, University of California, Berkeley (July 2004)
5. Clercq, D.D., Dimov, D.: Explaining venture capital firms' syndication behaviour: a longitudinal study. Venture Capital 6(4), 243–256 (2004)
6. Das, S.R., Jagannathan, M., Sarin, A.: Private equity returns: An empirical examination of the exit of venture-backed companies. Journal of Investment Management 1(1), 152–177 (2003)
7. Eldén, L.: Matrix Methods in Data Mining and Pattern Recognition, Fundamentals of Algorithms, vol. 4. SIAM, Philadelphia (2007)
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters 27(8), 861–874 (2006)
9. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1), 10–18 (2009)
10. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, 2nd edn. Springer Series in Statistics. Springer, Heidelberg (2009)
11. Livingston, F.: Implementation of Breiman's Random Forest Machine Learning Algorithm. Tech. rep., North Carolina State University ECE 591Q (Fall 2005), http://www4.ncsu.edu/~fjliving/docs/JournalPaper_Livingston.pdf
12. Milner, F., Vos, E.: Private Equity: a Portfolio Approach. Journal of Alternative Investments 5(4), 51–65 (2003)
13. Ueda, M., Hirukawa, M.: Venture Capital and Productivity. In: First Banca d'Italia/CEPR Conference on Money, Banking and Finance, Rome (October 2003), http://www.cepr.org/meets/wkcn/5/591/papers/ueda.pdf

A Rule-Based Method for Customer Churn Prediction in Telecommunication Services

Ying Huang, Bingquan Huang, and M.-T. Kechadi

School of Computer Science and Informatics,
University College Dublin,
Belfield Dublin 4, Ireland
ying.huang.1@ucdconnect.ie
{bingquan.huang,tahar.kechadi}@ucd.ie

Abstract. Rule-based classification methods, which provide the interpretation of a classification, are very useful in churn prediction. However, most of the rule-based methods are not able to provide the prediction probability which is helpful for evaluating customers. This paper proposes a rule induction based classification algorithm, called CRL. CRL applies several heuristic methods to learn a set of rules, and then uses them to predict the customer potential behaviours. The experiments were carried out to evaluate the proposed method, based on 4 datasets of University of California, Irvine(UCI) and one dataset of telecoms. The experimental results show that CRL can achieve high classification accuracy and outperforms the existing rule-based methods in churn prediction.

Keywords: Churn Prediction, Rule-based Classification, Hill-climbing Search, High Order Rules.

1 Introduction

Service companies in telecommunications suffer from a loss of valuable customers to competitors; this is known as customer churn. In the last few years, there have been many changes in the telecommunications industry, such as, new services, technologies and the liberalizations of the market increasing competition. Recently, data mining techniques have been emerged to tackle the challenging problem of customer churn in telecommunication service fields [1,2,3,4]. As one of the important measures to retain customers, churn prediction has been a concern in telecommunication industry and research.

As one of the important phases in churn prediction process, classification can directly affect the prediction results. Many modelling techniques have been successfully used for the classification, including Decision Tree (DT), Logistic Regression (LR), Artificial Neural Networks (ANN), Naive Bayes (NB) and Support Vector Machine (SVM). However, they still have some limitations. Most of the modelling techniques provide either prediction probability or classification rules. For example, NB, LR [5,6,7], ANN [8,9] and SVM [10,11] can only provide the probability of classification, whereas they do not produce decision rules. DT

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 411–422, 2011.

[12,13,14] can produce rules, however, the probability of classification cannot be obtained from DT model[15].

A number of Genetic Algorithm (GA) based classification models also have been reported in[16,17,18]. These models can be efficient. However, they have high computational complexity. Recently, DMEL (Data Mining by Evolutionary Learning) [15], a GA based prediction approach was proposed. This technique not only provides classification rules, but also gives the probability of classification. It was shown that DMEL is a very promising modelling technique, however, its computational complexity is too high.

This paper proposes a new rule-based classification method named CRL (**C**lassification by **R**ule **L**earning). CRL can not only classify samples by a set of generated rules, but also it is able to produce the prediction probability. Moreover the processing time of this method is reasonable. CRL consists of two main procedures: Firstly, generating rules by applying the heuristic of hill-climbing and pruning unimportant and redundant rules. Secondly, predicting one of the categories to which a test case belongs according to the rules. In the experiments, Lift Curve and AUC, were applied to demonstrate the performance of algorithm.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 describes the details of the proposed algorithm. Section 4 provides the experimental results and discussion. The conclusion and future work are given in Section 5.

2 Related Work

Rule-based classification is a well known domain. One of the main characteristics of rule induction is that they are more transparent and understandable than some other training models. A significant amount of research effort has been put into this domain. In a rule-induction process, usually two strategies are applied: general-to-specific (top-down) and specific-to-general (bottom-up). Due to that the strategy of general-to-specific is more widely applied in rule learning, this section will introduce several approaches by using it, besides examples of rule induction by applying the strategy of specific-to-general can be found in [19,20,21].

Among all the different methods for constructing the rules, decision tree based algorithms are the most popular [12,13,14]. C4.5 has proved to be one of the most well known among all tree-based classifiers. A tree is constructed using the method of divide-and-conquer. C4.5 starts to search an attribute with the highest information gain, and then the data is partitioned into classes according to this attribute. Each sub-tree represents a class, and will be recursively further partitioned using the same policy until some stopping criteria are satisfied, such as the leaf node has been reached etc. The rules can be obtained by traversing each branch of the tree. The details of this algorithm can be found in [12].

CN2 [22] works in an iterative way, each iteration searches one rule that *covers* (see section 3.1 for the definition) a large number of training cases by making use of beam search method. Entropy and likelihood ratio are used to evaluate

the quality and the significance of the rules. Rules are ranked in accordance with the quality of rules. When classifying a test case, CN2 goes through each rule until finding a rule that covers the case, then assigns the class label of that rule to the case.

DMEL is a genetic classification technique. The DMEL classifier consists of a set of labelled rules which were found by rule induction technique. DMEL starts by generating $first\_order$ rules, then $high\_order$ rules can be constructed iteratively by randomly combining lower order rules. Given a sample without labels, the DMEL classifier applies the rules to model the sample and makes the classification decisions. DMEL has been shown to be an efficient classifier. The details of this technique can be found in [15].

3 Algorithm of CRL

Although experimental results from [15] showed that DMEL is efficient at detecting churners, there is still improvement that can be done for better classification results. The objective in designing CRL is to improve the classification accuracy. CRL does so by considering several aspects: Firstly, CRL applies different heuristic methods for constructing classification rules; Secondly, constraints were introduced to avoid bad rules, by applying better strategies.

3.1 Basic Concepts

The Rules are of the following form: IF *antecedent* THEN *consequence*, where *antecedent* describes a conjunction of a certain number of <attribute, interval> pair, and *consequence* denotes the class label. Another basic concept of rule induction is *Cover*, which expresses the case that a sample satisfies the antecedent part of the rule. A rule *Correctly Covers* a sample when it correctly predicts the covered sample. The task of rule induction is to build a model by generating a set of rules, which either do or do not Cover cases.

general rule and $specific\ rule$ are two important concepts. Usually, a rule with a smaller number of <attribute, interval> pairs in the *antecedent* part is more *general* than the one having more. The reverse case is more $specific$. *most general* occurs when the number of antecedent is 1, while *most* $specific$ means the number of antecedent is the total number of attributes. Normally, the most specific rules should not be considered since they are too specific and not able to describe a relative big data space.

3.2 Rule Learning

CRL is a general-to-specific strategy based method, so it starts by generating the most general rules, first-order rules, then iteratively constructing higher-order rules based on the lower-order rules. Algorithm 1 illustrates the process of learning rules. SET_{rules} is the set of generated rules, k denotes the number of orders, and *threshold* denotes the maximal number of orders.

Algorithm 1. CRL Algorithm(training data)

```
1: SET_rules ← φ;
2: k ← 1;
3: SET_rules ← First_order_rules(training data);
4: while k <= threshold do
5:    high_order ← Higher_order_rules(lower_order_rules);
6:    SET_rules ← SET_rules ⋃ high_order;
7:    k++;
8: end while
9: return SET_rules
```

First-order Rules. Algorithm 2 illustrates the process of generating the first-order rules. In order to avoid information loss, all possible combinations of <attribute, interval> pair were taken into account. In Algorithm 2, $num\_attribute$, $num\_interval$ and num_class respectively denotes the number of attributes, the number of intervals of the current attribute and the number of different categories. Initially, the set of $first\_order\_rule$ is empty, $rule_{ijk}$ expresses a rule having the form "IF$f_i = Interval_j$ THEN $Class_k$", if Pruning (rule$_{ijk}$) returns false, then $rule_{ijk}$ can be accepted as a useful rule (Rule pruning is described in section 3.3).

Algorithm 2. $First\_order\_rules$(training data)

```
1: First_order_rules ← φ;
2: for i = 1; i <= num_attribute; i++ do
3:    for j = 1; j <= num_interval; j++ do
4:       for k = 1; k <= num_class; k++ do
5:          if Pruning(rule_ijk) returns false then
6:             First_order_rules ← First_order_rules ⋃ rule_ijk;
7:          end if
8:       end for
9:    end for
10: end for
11: return First_order_rules
```

High-order Rules. Higher-order rules are iteratively constructed based on the previous set of lower-order rules. The second-order is grounded on the $first\_order$, the $third\_order$ is based on the $second\_order$. Generally, $(n-1)\_order$ rules are the base for the n_order rules. Algorithm 3 illustrates the procedure of constructing a set of $high\_order$ rules. Positive lower order rules, which have the positive prediction, and negative lower order rules, which reversely have the negative prediction, can be separately obtained from the $lower\_order\_rules$. In algorithm 3, $item$ is used to define a <attribute, interval> pair. $positive\_items$ consist of all exclusive items extracted from all positive rules. We generate negative-items in the same way. Each positive rule becomes more specific by using the $hill\_climbing$ strategy to add one $item$ into the antecedent part of

the current rule. Algorithm 4 describes the heuristic of *hill_climbing* for constructing one high order rule. It will be discussed in detail. The rule pruning is also an important step for generating *high_order* rules. In order to reduce computation time, we get rid of rules having the same quality. Many methods can be used to assess the quality of rules. They either focus on the coverage or on the accuracy. However, a measure having the ability to make a good trade off between coverage and accuracy is more suited for rule learning. Therefore, Weighted Relative Accuracy(WRA) [23], which can be calculated by Eq. (1) is applied.

$$WRA = \frac{Num\_(a)}{Num\_total} \times (\frac{Num\_(a,c)}{Num\_(a)} - \frac{Num\_(c)}{Num\_total}) \quad (1)$$

where $Num\_(a)$ and $Num\_(c)$ represent the number of data cases that are covered by the antecedent and the consequence of a rule, respectively. $Num\_(a,c)$ is the number of cases correctly covered by a rule, and Num_total denotes the total number of cases in the training set. Line 9 expresses that only rules with exclusive quality can be added.

Algorithm 3. *Higher_order_rules*(a set of *lower_order_rules*)

```
1: all_rules ← φ;
2: positive_items ← get all exclusive items from positive_lower_order_rules;
3: negative_items ← get all exclusive items from negative_lower_order_rules;
4: accuracy_list ← φ;
5: for i = 1; i <= num_positive_rules; i++ do
6:    high_rule ← hill_climbing(positive_i, positive_item);
7:    if Pruning(high_rule) returns false then
8:       WRA_i ← calculate the Weighted_Relative_Accuracy for high_rule;
9:       if WRA_i is not in the list of accuracy_list then
10:         all_rules ← all_rules ⋃ high_rule;
11:         accuracy_list ← accuracy_list ⋃ WRA_i;
12:      end if
13:   end if
14: end for
15: do the same work to generate negative_rules;
16: all_rules ← all_rules ⋃ negative_rules;
17: return all_rules;
```

Algorithm 4 illustrates the process of generating one *high_order* rule. Let *accuracy_list*, which is a list consisting of different WRA values, be empty in the beginning. Basically, one piece of *high_order* is based on one piece of *lower_order* rule and a *new item*, which has been mentioned earlier. If the *new item* does not occur in the antecedent of the *lower_order* rule, then *one_high_order* can be built by combining *one_lower_rule* and the *new item*.

Meanwhile, the WRA value of the newly built *one_high_order* is stored in the *accuracy_list*. This algorithm returns one of the *high_order* rules with the highest WAR value.

Algorithm 4. *hill_climbing*(*one_lower_rule*, *all_low_items*)

```
1: accuracy_list ← φ;
2: for i = 1; i <= num_low_items; i++ do
3:    if one_lower_rule does not include item_i then
4:       one_high_rule ← combination of one_lower_rule and item_i;
5:       WRA_i ← calculate Weighted_Relative_Accuracy for one_high_rule;
6:       accuracy_list ← accuracy_list ⋃ WRA_i;
7:    end if
8: end for
9: BEST ← one of high order rules having the highest WRA;
10: return BEST
```

3.3 Pruning Rules

The number of generated rules can be huge, to make the classification effective, we must prune bad rules with insignificant or noisy information. Several criteria are used for pruning rules. Usually, χ^2 statistic test is used to test if there exists strong relative relationship between two attributes. In this paper, χ^2, which can be calculated by Eq. (2), is applied to decide whether a rule is significant which means all the attributes occurring in antecedent part of the rule correlate to the consequence.

$$\chi^2 = \frac{(O-E)^2}{E} \tag{2}$$

where O and E are the observed and expected frequency, which can be expressed by Eq. (3):

$$O = Num\_(a,c) \qquad E = \frac{Num\_(a) \times Num\_(c)}{Num\_total} \tag{3}$$

In addition to χ^2, $Support$ and $Confidence$ are another two essential factors when deciding if a rule should be pruned or not. $Support$ and $Confidence$ can be calculated by Eq. (4).

$$Support = \frac{Num\_(a,c)}{Num\_total} \qquad Confidence = \frac{Num\_(a,c)}{Num\_(a)} \tag{4}$$

Algorithm 5 illustrates the pruning method. α is the critical value for χ^2 significance test, and it is set to be 3.84 in this research. $minS$ and $minC$ are two threshold values, which define the minimal value of support and confidence, respectively. In this research, the minimum value of support and confidence were set to be 0.01 and 0.5, respectively. A rule can be retained if it satisfies all the threshold values.

Algorithm 5. Pruning(*rule*)

1: $Flag \leftarrow$ true;
2: **if** chi-square statistics $> \alpha$ AND $Support\_rule >$ minS AND $Confidence\_rule >$ minC **then**
3: $\quad Flag \leftarrow false$;
4: **end if**
5: **return** Flag;

3.4 Classification

This section discusses how to classify cases based on a number of classification rules. Rules consist of two categories, *churn* and *non_churn*. Two prediction models can be built based on all *churn* rules and all *non_churn* rules, respectively. In order to estimate the importance of each rule, we rank rules in each model based on a set of principles, which are also used in [24] and can be described as follows:

1. If $confidence\_1 > confidence\_2$, then $rule\_1$ has higher priority than $rule\_2$;
2. If $confidence\_1 = confidence\_2$ and $support\_1 > support\_2$, then $rule\_1$ also has higher priority than $rule\_2$;
3. If $confidence\_1 = confidence\_2$, $support\_1 = support\_2$ and $rule\_1$ is more general than $rule\_2$, then $rule\_1$ has higher priority than $rule\_2$.

After ranking, each rule has a *position* in each prediction model, one is more important than another rule if it has a better position than the other. We define the significance of each rule as follows:

$$Significance\_level = \frac{Num\_rules - position}{Num\_rules} \tag{5}$$

where Num_rules is the number of rules, *position* denotes the rule index of the ranked rule set.

Therefore, in a prediction model, the $Significance\_level$ of the most important rule is 1, while that of the least important one is $\frac{1}{Num\_rules}$. To classify a case, finding all covered rules from the churn model and *non_churn* model, respectively, if the sum $Significance\_level$ of covered rules in churn model is greater than that of *non_churn*, then assign *churn* to the case as the class label, otherwise, *non_churn* will be assigned. Apart from the mentioned two cases, if the current data case had the same sum $Significance\_level$ for churn model and *non_churn*, the majority class label should be assigned.

In order to know which customers are more likely to churn, CRL scores each customer according to the sum of its $Significance\_level$. If one is predicted to *churn*, then this customer is scored the sum of $Significance\_level$ of the churn model. Otherwise, the sum of $Significance\_level$ in *non_churn* model will be used to score that individual. Customers are sorted separately according to the prediction, cases predicted to churn are sorted in descending order, while cases

predicted to stay (non_churn) are sorted in ascending order. For both of the sorted lists, customers close to the top are more likely to become churners.

4 Experiments and Discussion

Experiments were conducted based on four UCI repository datasets [25] and a telecommunication dataset (six different class distributions) of the Ireland Telecoms [26]. The datasets are described in Table 1. Each of these datasets was equally divided into one training dataset and one testing dataset, they have the same distribution.

Table 1. Dataset descriptions

Dataset Names	# Instances	# attributes	Class Distribution
German Credit Data(UCI)	1000	20	30% : 70%
Heart Disease(UCI)	270	13	44.44% : 55.56%
Pima Indians Diabetes(UCI)	768	8	34.89% : 65.11%
Japan Credit Screening(UCI)	569	14	44.5% : 55.5%
Telecommunication 1	2181	121	1% : 99%
Telecommunication 2	2203	121	2% : 98%
Telecommunication 3	2249	121	4% : 96%
Telecommunication 4	2297	121	6% : 94%
Telecommunication 5	2300	121	8% : 92%
Telecommunication 6	2300	121	10% : 90%

4.1 Evaluation

In order to evaluate the classification performance, $Lift\ Curve$ measure [27] is used. The $Lift\ Curve$ measure is calculated by scoring each data case, sorting cases according to their scores, calculating two parameters that indicate the percentage of samples and the percentage of True Positive, respectively. The two parameters can be calculated by Eq. (6):

$$X = \frac{Num\_Samples}{Num\_Total} \qquad Y = \frac{Num\_True\_Positive}{Num\_Total\_Positive} \tag{6}$$

where $Num\_Samples$, Num_Total, represent the number of sample data, the total number of testing cases, respectively. Let $Num\_True\_Positive$ and $Num\_Total\_Positive$ be the number of samples that have been identified as positive, and the total number of positive cases in the whole testing dataset, respectively.

Occasionally, it is difficult to use lift curve to evaluate the detected percentages from different prediction modelling techniques or different feature subsets of data. To overcome this problem, we use the area under Lift Curve (AUC) [28] to evaluate the models. The area under a Lift Curve can be calculated by the following equation:

$$AUC = \frac{S_0 - n_0 \times (n_0 + 1) \times 0.5}{n_0 n_1} \tag{7}$$

where S_0 is the sum of the ranks of the class 0 (churn) test patterns, n_0 is the number of patterns in the test set which belong to class 0 (churn), and n_1 is the number which belongs to class 1 (nonchurn). The details of AUC can be found in [27,28].

4.2 Discussion

Figure 1 shows the experimental results, which compare the prediction performance between the new proposed classification method and DMEL. Each graph is generated by a different churn rate of training and testing dataset. In order to reduce costs, telecommunication operators randomly select(contact) a small amount of customers to offer special consideration or attractive services.

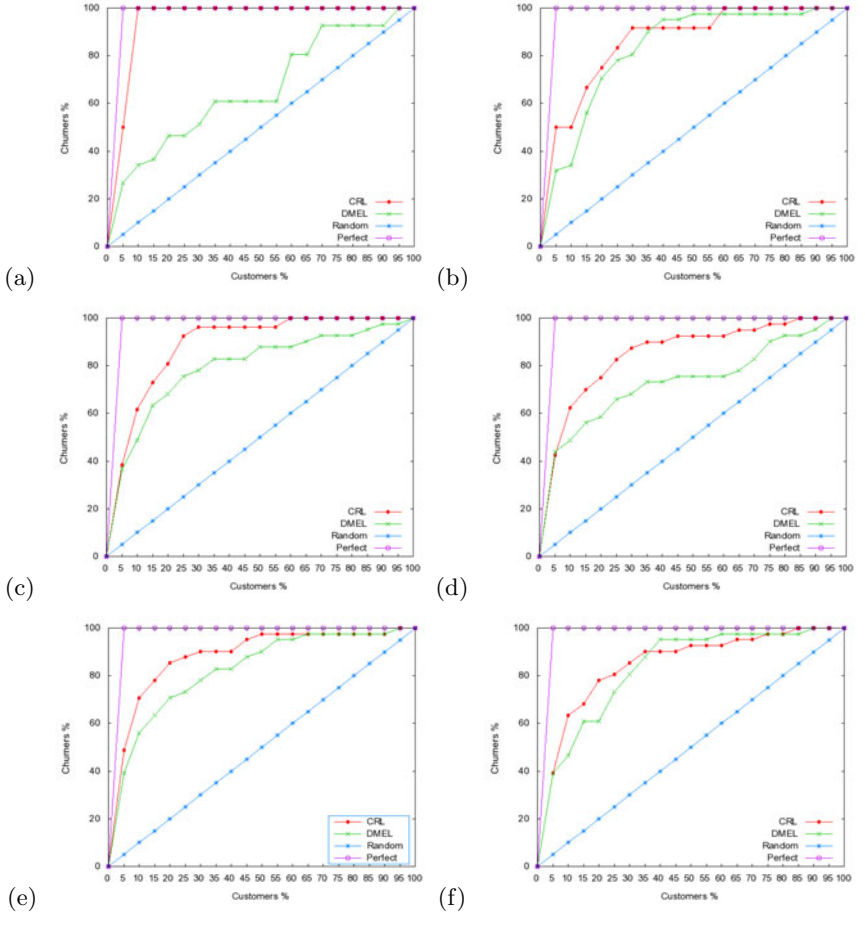


Fig. 1. Comparison of Lift Curves between CRL and DMEL, each sub-figure has different churn rate, (a):1%, (b): 2%, (c): 4%, (d): 6%, (e): 8%, (f): 10%

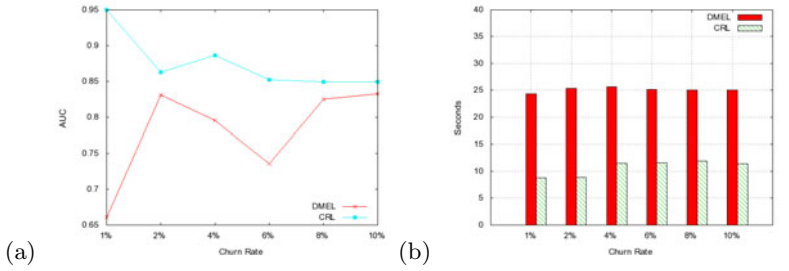


Fig. 2. (a): Compare of AUC values between CRL and DEML on different churn rate, (b): Comparison of time between CRL and DMEL

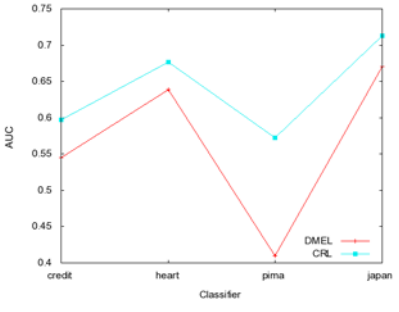


Fig. 3. Compare of AUC values between CRL and DMEL on a set of UCI data

The objective is catching relative more potential churners by contacting a relative small amount of customers. In Figure 1, the horizontal axis represents the percentage of contacted customers, while the vertical axis is the percentage of identified churners under the contacted customers. We can clearly see that the results based on CRL are more effective than DMEL. The lift curves generated by CRL always look better than those of DMEL for all percentages of contacted customers when the churn rates are 1%, 4%, 6% and 8%. When the churn rates are 2% or 10%, the lift curves representing CRL are not always higher than those of DMEL. We still regard CRL as the better algorithm given that it is not appropriate to contact so large a portion of customers when seeking to reduce costs for a telecommunication operator.

Moreover, the result of AUC measurement is consistent with that of Lift Curve. Figure 2(a) shows that there is a big difference between the AUC values of CRL and DMEL when the churn rates are 1%, 4% and 6%. For the rest of the cases, there still exists obvious difference. Figure 2(b) displays the precessing time when using the two algorithms. It shows that the proposed method is quicker than DMEL.

In order to show that CRL can be extended to other data collections, several sets of artificial data were tested. Figure 3 compares the AUC values between

the proposed classification algorithm and DMEL on four sets of UCI data. In Figure 3, the horizontal axis denotes different data set, while the vertical axis shows the obtained AUC values based on the two algorithms. It shows that the proposed algorithm produced better classification for each data set.

5 Conclusion and Future Works

In this paper, we proposed a new rule learning based classification algorithm. The experiments were conducted based on a telecommunication data sets and 4 other data-sets. The Lift Curve and AUC techniques are used to evaluate the performance of the algorithms. The experimental results show that CRL is more efficient than DMEL.

The classification results slightly vary when setting different values for the order to generate rules. Thus, it is necessary to find a proper threshold to control the number of high order rules rather than setting the size arbitrarily. In addition, the minimum values of support and confidence should also be considered more in the future since these will impact on the effectiveness of rule pruning.

References

1. Wei, C., Chiu, I.: Turning telecommunications call details to churn prediction: a data mining approach 23, 103–112 (2002)
2. Hung, S.-Y., Yen, D.C., Wang, H.-Y.: Applying data mining to telecom churn management. Expert Systems with Applications 31, 515–524 (2006)
3. Huang, B.Q., Kechadi, M.-T., Buckley, B.: Customer churn prediction for broadband internet services. In: Pedersen, T.B., Mohania, M.K., Tjoa, A.M. (eds.) DaWaK 2009. LNCS, vol. 5691, pp. 229–243. Springer, Heidelberg (2009)
4. Huang, B.Q., Kechadi, M.-T., Buckley, B.: A new feature set with new window techniques for customer churn prediction in land-line telecommunications. Expert Systems with Applications 37, 3657–3665 (2010)
5. Williams, D., Liao, X., Xue, Y., Carin, L.: Incomplete-data classification using logistic regression. In: ICML, pp. 972–979. ACM Press, New York (2005)
6. Komarek, P., Moore, A.A.: Logistic regression for data mining and high-dimensional classification. Technical report (2004)
7. Lemeshow, S., Hosmer, D.W.: Applied logistic regression
8. Penny, W.D., Roberts, S.J., and London Sw Bt: Bayesian neural networks for classification: How useful is the evidence framework (1998)
9. Bernatzki, A., Eppler, W., Gemmeke, H.: Interpretation of neural networks for classification tasks. In: Proceedings of EUFIT, pp. 1420–1424 (1996)
10. Dinggang, S., Yiqiang, Z.: Design efficient support vector machine for fast classification. IEEE Transactions on Neural Neworks 11, 124–136 (2000)
11. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2(2), 121–167 (1998)
12. Ross Quinlan, J.: C4.5: Programs for Machine Learning, 1st edn. Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann, San Francisco (January 1993)
13. Quinlan, J.R.: Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research 4, 77–90 (1996)

14. Du, W., Zhan, Z.: Building decision tree classifier on private data (2002)
15. Au, W., Chan, C.C., Yao, X.: A novel evolutionary data mining algorithm with applications to churn prediction. IEEE Transactions on Evolutionary Computation 7, 532–545 (2003)
16. Kelly, J.D.: A hybrid genetic algorithm for classification. In: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, pp. 645–650. Morgan Kaufmann, San Francisco (1991)
17. Kwedlo, W., Kretowski, M.: Discovery of decision rules from databases: An evolutionary approach. In: Żytkow, J.M. (ed.) PKDD 1998. LNCS, vol. 1510, pp. 370–378. Springer, Heidelberg (1998)
18. Pei, M., Goodman, E.D., Punch Iii, W.F., Ding, Y.: Genetic algorithms for classification and feature extraction. In: Annual Meeting, Classification Society of North America (1995)
19. Domingos, P.: Rule induction and instance-based learning: A unified approach, pp. 1226–1232. Morgan Kaufmann, San Francisco (1995)
20. Domingos, P.: Using partitioning to speed up specific-to-general rule induction. In: Proceedings of the AAAI 1996 Workshop on Integrating Multiple Learned Models, pp. 29–34. AAAI Press, Menlo Park (1996)
21. Domingos, P.: Efficient specific-to-general rule induction. AAAI Press, Menlo Park (1996)
22. Clark, P., Niblett, T.: The cn2 induction algorithm. Machine Learning, 261–283 (1989)
23. Todorovski, L., Flach, P.A., Lavrač, N.: Predictive performance of weighted relative accuracy. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 255–264. Springer, Heidelberg (2000)
24. Li, W., Han, J., Pei, J., and Class association Rules: Cmar: Accurate and efficient classification based on multiple class-association rules (2001)
25. http://archive.ics.uci.edu/ml/datasets.html
26. http://www.eircom.ie/cgi-bin/bvsm/mainPage.jsp
27. Vuk, M.: Roc curve, lift chart and calibration plot (2006)
28. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognition 30, 1145–1159 (1997)

Adaptive and Effective Keyword Search for XML*

Weidong Yang, Hao Zhu, Nan Li, and Guansheng Zhu

School of Computer Science, Fudan University
Shanghai, China

Abstract. Most of the existing methods for XML keyword search are based on the notion of Lowest Common Ancestor (LCA). However, as we explore the most important fundamental flaw inside those result models is that the search results are eternally determined and nonadjustable. In order to serve better results, we propose a novel and flexible result model which can avoid all these defects. Within our model, a scoring function is presented to judge the quality of each result. The considered metrics of evaluating results are weighted, and can be updated as needed. Based on the result model, three heuristic algorithms are proposed. Moreover, a mechanism is employed to select the most suitable one out of these algorithms to generate better results. Extensive experiments show that our approach outperforms any LCA-based ones with higher recall and precision.

Keywords: Keyword Search, XML, LCA.

1 Introduction and Motivation

XML Keyword Search is a user-friendly information discovery technique, which attracts many interests these years. Different with keyword search over flat documents, the search object is a single XML document/database with structure information inside and the results are supposed to be fragments of it containing keywords. Since it is difficult and sometimes impossible to identify users' intentions through keywords, it is indeed a difficult task to determine which fragments should be returned. Many valuable models are proposed to define the results, and the most popular ones are the Smallest Lowest Common Ancestor (SLCA) model [11] and its variations [2],[3],[4],[6],[7],[9],[10],[12], all of which are called as the *LCA-based models* in this paper. These researches regard the XML document as a rooted, labeled, unordered tree in which each inner node is an element or an attribute and each leaf is a value which may contain some keywords. In SLCA model a result is defined as a subtree that: (1) the labels of whose nodes contain all the keywords, (2) none of its subtree satisfies the first condition except itself. The root of such a subtree is called a SLCA node. It's recognized that SLCA model is definitely not a perfect one. Later works like [4],[5],[6],[12] illustrate that some results of the SLCA method are meaningless and some meaningful ones are missing by giving different examples, which are called the *false positive* and

* This research is supported in part by the NSF of China under grant 60773076, the Key Fundamental Research of Shanghai under Grant 08JC1402500, Xiao-34-1, 863 Program of China under Grant 2008AA121706.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 423–434, 2011.

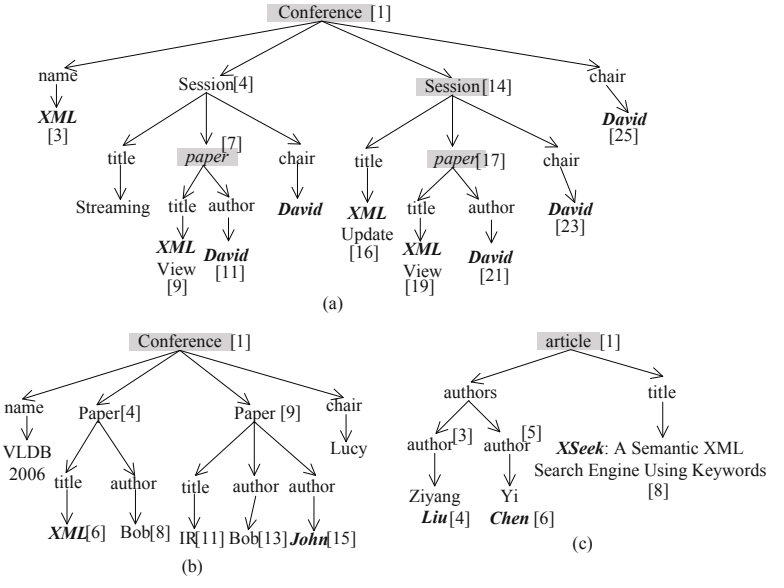


Fig. 1. Original examples from LCA-based approaches

the *false negative* problems respectively. Interestingly, the remedy approaches they propose sometimes conflict with each other, and counterexamples can always be found to testify that the two problems still happen.

In order to explain these issues we employ the original examples from former researches [4],[6],[12], which are illustrated in Figure 1 as three separate XML trees. In these trees keywords are marked in bold and important nodes are identified by numbers. Besides, we use $node_i$ to denote the node with the number i in any of the trees. Consider keywords {"XML", "David"} issued on the XML document in Figure 1(a), apparently SLCA method can find two subtrees rooted in $node_7$ and $node_{17}$. In many cases the subtrees are not appropriate for users because their sizes are too large and plenty of meaningless information is involved. GDMCT [3] approach proposes a good way to handle this that it returns the Minimum Connecting Trees (MCTs) instead. A MCT is defined as a subtree which employs the SLCA node as root and the keyword nodes as leaves. Since the refining of the final results is not a focus in this paper, for convenience a search result is considered as a group of keyword nodes instead of a document fragment by us. In this example, SLCA method finds two results $\{node_9, node_{11}\}$ and $\{node_{19}, node_{21}\}$ which are two separate papers in semantics.

MLCA method [7] and GDMCT method [3] should agree with the answers because they both define a result model very similar to SLCA. But, ELCA [2][12] method will claim that several reasonable results are missed. For instance, $\{node_3, node_{25}\}$ and $\{node_{16}, node_{23}\}$, which indicate a conference and one of its sessions respectively and are perfectly exclusive to each other, along with two SLCA results, are regarded as qualified ELCA results. However, neither $node_1$ nor $node_{14}$ is a SLCA node due to at least one SLCA node is their descendant node.

The semantics of Exclusive Lowest Common Ancestor (ELCA) is firstly proposed in [2] although it is not named as this. Afterwards, Xu et al. [12] and Zhou et al. [14] provide more efficient algorithms to retrieve the ELCA nodes. The formal definition of ELCA is complicated, however all the ELCA nodes can be retrieved through a straightforward two-step process which can help understanding the concept: (1) find all the SLCA nodes, halt the process if there isn't any; (2) remove all the SLCA nodes along with the subtrees rooted in them, then turn to the first step. The union of all the SLCA nodes obtained each time in the first step is indeed the set of ELCA nodes. Obviously, ELCA method can obtain more reasonable results.

Seemingly ELCA model has fixed the false negative issue of SLCA model and thus can find results perfect enough. However, based on the example illustrated in Figure 1(b), Li et al. [6] claim that both SLCA and ELCA models suffer from the false positive problem. Suppose {"XML", "John"} are the keywords, either SLCA or ELCA method returns $\{node_6, node_{15}\}$ as the only result which is thought meaningless in [6]. Actually, more examples can be found to support this point of view because in some cases the keyword nodes in a result could be really far from each other in the tree. Rather than implement semantics inference to improve the results as XSeek [8] does, Li et al. introduce a simple rule to filter all the ELCA nodes. For any two keyword nodes n_i and n_j in an ELCA result, suppose $lca(n_i, n_j)$ is the LCA node of n_i and n_j and two nodes n_i' and n_i' are the ones in the paths "$lca(n_i, n_j) \rightarrow n_i$" and "$lca(n_i, n_j) \rightarrow n_j$" respectively satisfy that n_i' and n_j' have the same elementary type, then the result is unqualified. Accordingly, $\{node_6, node_{15}\}$ is not a qualified result because $node_4$ and $node_9$ have the same elementary type. After the filtering, the left ELCA nodes are called Valuable Lowest Common Ancestor (VLCA) nodes.

The rule of VLCA method is actually first proposed by Cohen et al. [1]. They use this to determine if two keyword nodes are "meaningfully related". Indeed it is kind of overstrict, and more relaxed criteria could be used such as the LCA have to be low or the result should have a good compactness. Kong et al. present a counterexample in [4]. To search the keywords {"Liu", "Chen", "XSeek"} in the tree from Figure 1(c), $\{node_4, node_6, node_8\}$ is a reasonable answer yet will be eliminated by VLCA method because $node_4$ and $node_5$ have the same elementary type "author". Kong et al. [4] also propose a concept called Related Tightest Fragments (RTF) as final search results, which is equal to representing the results of ELCA method in MCTs. Obviously, it keeps the vague problem of false positive results being existed.

Following the common sense that results should be returned as many as possible, we shouldn't care too much about the false positive issue actually. At least we can still return them to users with lower rank scores. On the other hand, another thing needs to be paid more attention to is how to evaluate the results properly and so that to judge if returned results are good enough or are there better ones can be found. No doubt out of the aforementioned models ELCA method can get the maximum number of results which is actually a superset of the results returned by any other LCA-based method. Next we use another example to discuss the ELCA results to explain the problem.

Example 1: Figure 2 illustrates another XML document in tree structure which stores the information of proceedings and journals in DBLP. There is a recursive situation

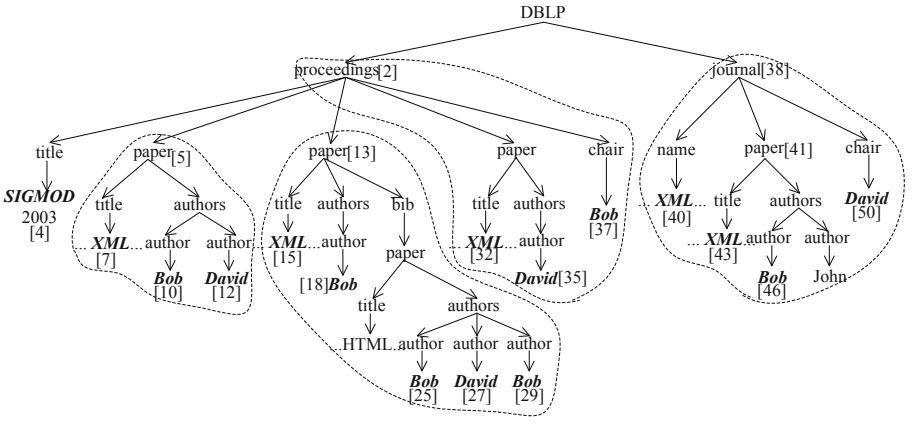


Fig. 2. An example of XML tree

in the document that a paper element could have a descendant which is also a paper. Given three keywords {"XML", "Bob", "David"}, there are four ELCA results could be found in the document which are illustrated in closed dashed curves. Apparently, $\{node_7, node_{10}, node_{12}\}$ is a very good answer which is a paper whose title is about "XML" and its authors are "Bob" and "David". From another point of view, the LCA node is appropriately low and the *compactness* (which is recognized as an important measure by many researches and can be calculated through dividing the number of keyword nodes by the number of all nodes in a result) is good. Another result $\{node_{15}, node_{18}, node_{25}, node_{27}, node_{29}\}$ consists of two connected papers and also can be regarded as a meaningful one. With respect to the other two results some people probably have different opinions. For $\{node_{32}, node_{35}, node_{37}\}$, although it satisfies the constraint VLCA model requires, it is hard to say that $node_{35}$ and $node_{37}$ are meaningfully related. For $\{node_{40}, node_{43}, node_{46}, node_{50}\}$, many may argue that it will be better if it is split into two separate results $\{node_{40}, node_{50}\}$ and $\{node_{43}, node_{46}\}$. Suppose another keyword "SIGMOD" is added into the query, there will be a single large result which can be represented by the subtree rooted in $node_2$ returned by ELCA approach. It is too large to be an appropriate result for users, yet it cannot be divided since any of its subparts (such as the three in the closed dashed curves) is considered totally meaningless because it doesn't contain all the keywords. Besides, the keyword nodes in the subtree rooted in $node_{38}$ can never be included in any result because of the same reason. The false negative problem still happens in a way.

Here we enumerate four defects of ELCA result model, which are actually the flaws of all the LCA-based models.

1. *A lack of universal criteria to judge whether a result is qualified or not.*
   VLCA model tries to point out the false positive problem of ELCA model through the specific example in Figure 1(b) and afterwards provides a strict rule to filter results. The practicability of the rule is simply disconfirmed by the RTF model

through another specific example. We don't think either of their arguments is convincing. In an extreme case, we can say that any fragment containing keywords is meaningful. However, to clarify which ones are qualified results and which ones are not, what we need is a quantifiable metric rather than some strict rules.

2. *Not enough features are considered.*
   In the ELCA model, a result is restricted to contain all the keyword nodes. Besides, only the LCA nodes of the results are considered. At least two important features are missed: the compactness and the size. Because the size doesn't mean anything in ELCA model (or any of other LCA-based models), large-size results keep showing. Although some works [2],[4],[5] consider several features in their ranking models, except for LCA nothing is taken into account in the process of retrieving the results.
3. *Some useful information is omitted in the results.*
   For instance in Example 1 when the keywords {"XML", "Bob", "David", "SIGMOD"} are used to search the document, any keyword node in the subtree rooted in $node_{38}$ is ignored despite they are more or less meaningful to users. Actually all the LCA-based models only serve the *best* results and refuse to organize and return the *second-best* ones. Although some works [5],[10] support the OR semantics and so that can find such information, this only can be utilized in a narrow way that users have to specify the AND/OR logic in their queries. In this example, users have to provide a query like "(XML AND Bob AND David) OR SIGMOD" even though they have no idea of what exactly the real data is.
4. *Despite different users probably have distinct intentions by using the same keywords, the search results are eternally determined and nonadjustable.*
   Check Example 1 again. When given keywords are {"XML", "Bob", "David", "SIGMOD"} we don't know which one is better: to return a proceeding with a lot of information or to generate smaller and compact papers. However with any LCA-based approach neither the administrator nor a user has a choose to improve their search results. Therefore, we need to provide a more flexible model in which we can adjust the results according to the context or the feedback from users.

In this paper, we propose a novel result model for XML keyword search which can avoid all these defects perfectly. Instead of applying restrictions upon the results we give each result a score to evaluate the quality of it. Such a score is generated by a scoring function considering sufficient features, and each is weighted so that can be adjusted as necessary. Obviously, a result gets a better quality if it has a higher score. Since the results could be distinct when the features are given different weights. We provide multiple algorithms to generate results and based on the values given to the parameters in the scoring function the most suitable algorithm will be chosen to serve better results. Each of our algorithms is supposed to find not only the results with the highest scores as many as possible but also those *second-best* ones with lower scores.

The remainder of the paper is organized as follows.Section 2 defines the novel model for XML keyword search results. The algorithms are proposed in Section 3. Afterwards, experimental results are exhibited in Section 4. Finally we conclude the paper in Section 5.

2 Result Model

In this section, we provide the formal definition of our search results. As explained before we use a set of keyword nodes to represent a final result for simplicity. Besides, rather than applying some restrictions upon the results, we give each result a score to evaluate its quality (how meaningful it is). Such a score of a keyword node set R is calculated through a function $score(R)$. R is a better result if it has a higher $score(R)$, and in our opinion any R satisfying $score(R) > 0$ can be regarded as a result. Unlike any LCA-based model, in which overlapping and inclusion are not allowed between two individual results, we hold a different opinion that we think two results can share some common nodes as long as none of them is a keyword node, which means our model is more relaxed to the results.

Definition 1: (**XML Keyword Search Results**) T is an XML document/databse which could be viewed as a tree. Given a set of t keywords $\{w_1, \ldots w_t\}$, K is the set of all the keyword nodes whose labels contain any keyword. Then, the result set of searching $\{w_1, \ldots w_t\}$ in T is $\mathbb{R}$ which satisfies:

- $\forall R \in \mathbb{R}$ is a set of keyword nodes that $score(R) > 0$;
- $\forall R_i, R_j \in \mathbb{R}$, $R_i \cap R_j = \emptyset$;
- $\bigcup_{R \in \mathbb{R}} R = K$.

From these three conditions in Definition 1 we can see that the XML tree is divided into separate fragments each of which is meaningful. In other words, XML keyword search can be regarded as a problem that dividing the set of all keyword nodes into groups that are meaningful. Clearly, excessive partitions of K can be qualified result sets in accordance with Definition 1 and we have to judge which ones are gratifying and which ones are not. However, comparing with the LCA-based models it has two advantages. First, each result is not given specific restrictions yet can be appraised. Second, all the keyword nodes are involved in the results which definitely brings a high recall. Next we define a concept of the *optimal result set* which is supposed to be a standard for our result-finding algorithms to pursue.

Definition 2: (**Optimal XML Keyword Search Result Set**) Suppose $\mathbb{R}$ is a partition set of K and $|\mathbb{R}| = n$. We arrange the items in $\mathbb{R}$ to be a sequence $S = \{R_1, ..., R_n\}$ which satisfies: for any $1 \leq i < j \leq n$, $score(R_i) \geq score(R_j)$. Then, $\mathbb{R}$ is called an optimal result set iff:

- $\forall R' \subset (R_k \cup ... \cup R_n)$, $1 \leq k \leq n$, $score(R_k) \geq score(R')$.

In other words, $score(R_1)$ is the biggest score can be found in the set of all the keyword nodes K, and $score(R_2)$ is the greatest score in the set $(K - R_1)$, and so on. The rationality of this definition is quite clear that users always prefer the best results and in our model the best results are those results with the largest scores.

A proper scoring function represents a well-designed evaluation model, in which at least four metrics need to be involved: (1) the content information, which mainly refers to the keywords it contains; (2) the structure information, which specifically means the hierarchical position of the root (the LCA of the keyword node set); (3) the

compactness, which usually can be calculated through dividing the number of keyword nodes by the number of all nodes in the fragment; (4) the size, which must not be too large. More importantly, the evaluation model should be adjustable to suit different contexts. As a matter of fact, in the evaluation model we can use as many features as possible when they are reasonable. Meanwhile, the LCA-based models only consider the first two features, the content and the structure information, and that's why they suffer from those defects aforementioned.

Some extra notations have to be defined as follows before the scoring function is proposed:

- K is the set of all the keyword nodes;
- for any node v, $dpt(v)$ returns the depth of v in the tree (the depth of the root is 1, and the height of the tree is h);
- for any set of keyword nodes N, $kn(N)$ is the function to get the exact number of keywords N contains.
- for any set of keyword nodes N, $mct(N)$ is the set of all the nodes contained by the MCT of N.

For any result R we provide separate formulas to evaluate the four kinds of features a result possesses: $\mathscr{T}(R)$, $\mathscr{H}(R)$, $\mathscr{C}(R)$ and $\mathscr{S}(R)$. Each of the functions returns a real number between [0, 1], and the greater the value is R is better in one respect.

- Content Information: $\mathscr{T}(R) = kn(R)/t$;
- Structure Information: $\mathscr{H}(R) = dpt(lca(R))/h$;
- Compactness: $\mathscr{C}(R) = |R|/|mct(R)|$;
- Size:

$$\mathscr{S}(R) = \begin{cases} 1 & |mct(R)| \leq st \\ 0 & |mct(R)| > st \end{cases}$$

In the formula of $\mathscr{S}(R)$, st is an integer which is the size threshold of R's MCT. When the size exceeds the threshold, the result is considered inappropriate to be returned. Finally, the scoring function is defined as follows.

$$score(R) = \frac{(1+\mathscr{T}(R))^{\alpha} \times (1+\mathscr{H}(R))^{\beta} \times (1+\mathscr{C}(R))^{\gamma} \times \mathscr{S}(R)}{2^{(\alpha+\beta+\gamma)}} \quad (1)$$

In formula (1) α, β, and γ are three adjustable parameters, each of which should be a positive real number greater than or equal to 1. Furthermore, according to former discussion under normal circumstances α should be set much greater than β, at the same time β is usually larger than γ. Hence we have $\alpha > \beta > \gamma \geq 1$. It is easy to find that any $score(R)$ is either equal to zero or between $(1/2^{(\alpha+\beta+\gamma)}, 1]$.

3 Algorithms

For an XML tree and given keywords, to find the optimal result set is certainly the ultimate goal of us. Nevertheless, there is a lack of an efficient way to achieve this purpose due to a complicated situation that any feature in the scoring function is allowed

Algorithm 1. $Matrix(K)$

```
1: transform K to ℂ;
2: build the score matrix M for ℂ
3: while there is a positive item in M
4:     find the greatest m_ij in M;
5:     remove C_i and C_j from ℂ;
6:     add C_i ∪ C_j into ℂ;
7:     update M according to current ℂ;
8: return ℂ;
```

to be weighted casually. In a naive approach, we have to enumerate all the possible subsets of K to find the result with the greatest score which alone has a time complexity of $O(2^n)$. Therefore, in this section we propose several heuristic algorithms with high efficiency to generate results hoped to be close to the optimal one. Especially, some of them are developed to adapt to specific circumstances. In the next section, many experimental results show the effectiveness of these algorithms.

3.1 Matrix Algorithm

From Definition 1, the XML keyword search problem is equivalent to dividing the set of all the keyword nodes into groups that are meaningful, which is actually the clustering techniques are supposed to handle. In this subsection, we provide a basic agglomerative hierarchical clustering algorithm to obtain the result set $\mathbb{R}$. Before conduct the algorithm (or any algorithm we propose later) the nodes in the XML tree have been encoded by Dewey code and the inverted index of term-node has been built. So that, each time the calculation of a score costs $O(1)$.

The input of the clustering algorithm is the set of all the keyword nodes K which is then transformed to a candidate set $\mathbb{C}$ that each entry $C \in \mathbb{C}$ is a node set and originally contains an individual keyword node. Afterwards, a *Score Matrix* of $\mathbb{C}$ is built. Suppose $|K| = n$ and $\mathbb{C} = \{C_1, ..., C_n\}$, then the score matrix of $\mathbb{C}$ is an n-by-n matrix M that each item m_{ij} is set to be $score(C_i \cup C_j)$ if $score(C_i \cup C_j)$ is greater than both $score(C_i)$ and $score(C_j)$, otherwise m_{ij} is 0. At each step we find the highest m_{ij} in the matrix and merge C_i and C_j to be a new result in $\mathbb{C}$. Then, the matrix is updated to be a $|\mathbb{C}| \times |\mathbb{C}|$ one for current $\mathbb{C}$. The program stops when there is no positive value left in the matrix, then $\mathbb{C}$ is the final result.

Such an algorithm is called the Matrix algorithm and is shown in Algorithm 1. As we know the space complexity of this algorithm is $O(n^2)$, and in the worst case the time complexity is $O(n^3)$. If the scores are stored as sorted lists (or heaps), the time complexity is reduced to $O(n^2 logn)$. Still it is not as good as the performance of the algorithms from the LCA-based approaches. However, theoretically it generates much better results because it calculates possible scores as many as possible and always chooses the largest one.

Algorithm 2. $CIF(\mathbb{K})$

```
1: while 𝕂 ≠ ∅
2:   m = |𝕂|;
3:   set K_1 as the set with the smallest size in 𝕂;
4:   if (m == 1)
5:     ℝ = ℝ ∪ clustering(K_1);
6:     break;
7:   else
8:     remove K_1 from 𝕂;
9:     transform K_1 to semi-results 𝕊;
10:    for (2 ≤ i ≤ m);
11:      transfer one or more nodes from K_i into each set in 𝕊;
12:      if K_i = ∅ remove K_i from 𝕂;
13:    ℝ = ℝ ∪ 𝕊;
14: return ℝ;
```

3.2 Content-Information-First (CIF) Algorithm

Many believe that the Content information should be an overwhelming criterion to evaluate a result and thus in our scoring function they would prefer α to be much larger than β and γ. Under this circumstance, the results containing all the keywords should be returned as many as possible and a results with insufficient content information R will only be generated in two cases. First, there is no result R' can be found that satisfies: (1) $R \cup R'$ contains all the keywords; (2) $score(R \cup R') > score(R')$. Second, such a R' can be found however the size of $R \cup R'$ exceeds the limit. Among these results with insufficient content information one definitely dominates another when it contains more keywords. In this subsection, we present an algorithm called the Content-Information-First (CIF) algorithm to retrieve such results.

Given a set of t keywords $\{w_1, ..., w_t\}$ there are t sets of keyword nodes $K_1, ..., K_t$ possessed through the inverted index and each K_i stores all the keyword nodes containing the keyword w_i. Let K_1 be the one with the smallest size. In line with the principle that a keyword node only exists in a single result, we can only obtain $|K_1|$ results containing all the keywords at most. Thus, we employ a program that for each set from K_2 to K_t we distribute one or more keyword nodes in it to every node in K_1 which can form a result with highest score. We use an example to explain the details before presenting the CIF algorithm.

Let $\mathbb{K} = \{K_1, ..., K_t\}$ and $\mathbb{R}$ to be the set of results which is empty originally, then the pseudo-code of CIF algorithm is illustrated in Algorithm 2. In the best case (when the sizes of the keyword node sets are even) the time complexity of CIF is $O(|K_1| \times \sum_{2 \le i \le t}(|K_i|))$, and in the worst case the time complexity is $O(|K_1| \times \sum_{2 \le i \le t}(|K_i|^2))$.

3.3 Structure-Information-First (SIF) Algorithm

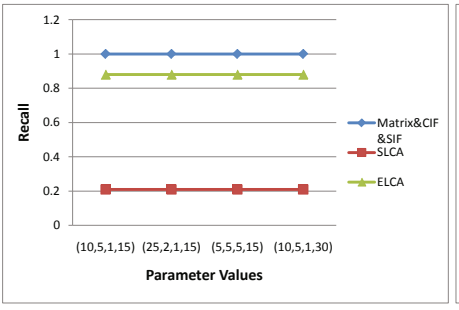
In some specific cases, we concern what the structure of a result much more than how much keyword information inside. For example, it is quite reasonable to assume that no matter how many keywords a result contains only those papers are qualified when proceeding keyword search on DBLP data set. In this case, we can set α close to or even smaller than β and γ and then generate results based on some restrictions built on the structure. Here we provide an algorithm called the Structure-Information-First Algorithm (SIF) which actually comes from another work of us [13] in which it is called the Core-driven Clustering Algorithm. To save space we don't explain the details here. Normally the time complexity of Algorithm SIF is O(n), and O(n^2) in the worst case.

4 Experiments

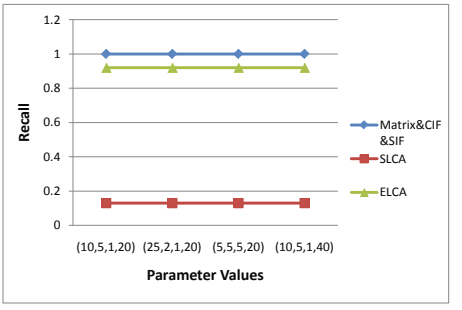
Extensive experiments are performed to compare our approaches with SLCA and ELCA approaches. For SLCA and ELCA approaches the Indexed Lookup Eager Algorithm [11] and the Indexed Stack Algorithm [12] are implemented respectively.

We use two metrics to evaluate them: recall and precision. Since we presume any keyword node is meaningful to users, the recall value can be simply calculated by dividing the number of the keyword nodes in the result set with the number of all keyword nodes in the document. The standard definition of precision from Information Retrieval is difficult to be followed here. Because in any search, each result returned is believed to be a satisfying one, which means they all consider the precision of their result set to be 1. Therefore, we design a variation called the proximity precision. Out of the set of results we find the one with the greatest score which is then considered as the best result and used as a standard . For a certain small number k, the top-k results with the largest scores are found and then an average score value is calculated which afterwards is divided by the greatest score to get the value of proximity precision.

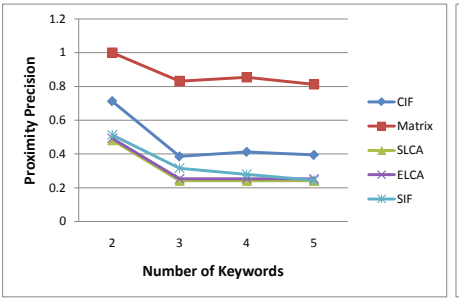
Figure 3(a) and Figure 3(b) show that the results generated by our approach have a overwhelming recall value comparing with SLCA and ELCA. We can see that SLCA always gets a poor value because lots of keyword nodes are abandoned. Our approach always has the largest recall since any keyword node is considered meaningful and included in a result. Figure 3(c) and Figure 3(d) illustrate how the proximity precision values change when we vary the number of keywords. The parameters are set to static values, and for any approach we only consider the scores of top-10 results. Obviously, Matrix and CIF overcome the other three. SIF doesn't act quite good since in the scoring function α is set much larger than β and γ. Similarly, when we give the parameters some static values and use three keywords to search each of the data sets, Figure 3(e) and Figure 3(f) show how the proximity precision values change as the value of k varies. There is a dramatic decline while k is enlarged for SLCA and ELCA. This can be explained that they usually only find a few best results and omit those second-best ones. In the last two graphs in Figure 3 the keyword numbers are both 4. We can see that when we change the values of parameters Matrix has a stable and excellent performance which is much better than SLCA and ELCA do. The most interesting thing is, for either CIF or SIF the proximity precision changes severely with different parameter values. That's why it is so important to select the appropriate algorithms according to them.



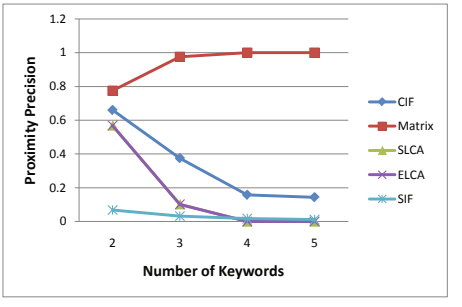
(a)Recall on DBLP,varying parameter values



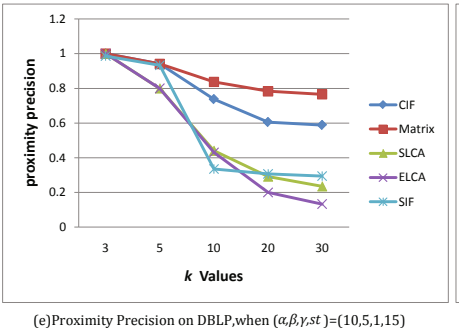
(b)Recall on TreeBank,varying parameter values



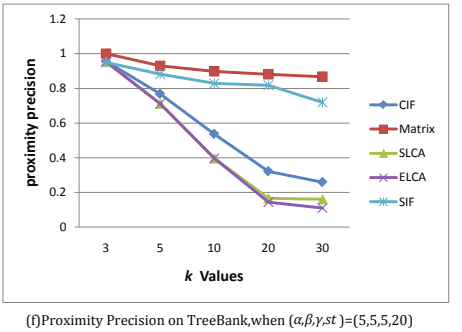
(c)Proximity Precision on DBLP,when(α,β,γ,st)=(10,5,1,15),k =5



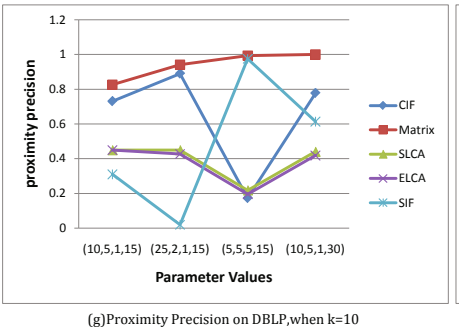
(d)Proximity Precision on TreeBank,when (α,β,γ,st)=(10,5,1,15),k =5



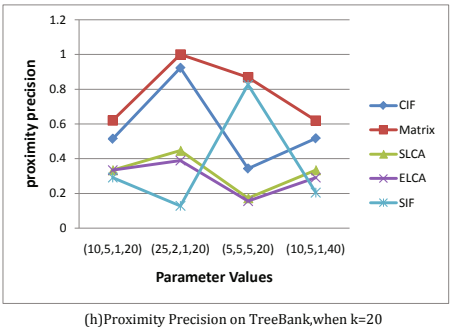
(e)Proximity Precision on DBLP,when (α,β,γ,st)=(10,5,1,15)



(f)Proximity Precision on TreeBank,when (α,β,γ,st)=(5,5,5,20)



(g)Proximity Precision on DBLP,when k=10



(h)Proximity Precision on TreeBank,when k=20

Fig. 3. Experimental Results

5 Conclusion

In this paper, several defects of LCA-based result models are explored. After that we propose a result model more effective and flexible which is mainly built upon a scoring function. Based on the novel model heuristic algorithms are provided to generate the keyword search results. Most importantly according to the given values of some parameters in the scoring function the most suitable algorithm can be automatically selected to generate the quality results efficiently. Finally, experimental results show that our approach outperforms any LCA-based ones with higher recall and precision.

References

1. Cohen, S., Mamou, J., Kanza, Y., Sagiv, Y.: XSEarch: A Semantic Search Engine for XML. In: Proceedings of the 29th International Conference on Very Large Data Bases (VLDB 2003), pp. 1069–1072 (2003)
2. Guo, L., Shao, F., Botev, C., Shanmugasundaram, J.: XRANK: Ranked Keyword Search over XML Documents. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (SIGMOD 2003), pp. 16–27 (2003)
3. Hristidis, V., Koudas, N., Papakonstantinou, Y., Srivastava, D.: Keyword Proximity Search in XML Trees. IEEE Trans. Knowl. Data Eng (TKDE) 18(4), 525–539 (2006)
4. Kong, L., Gilleron, R., Lemay, A.: Retrieving Meaningful Relaxed Tightest Fragments for XML Keyword Search. In: Proc. 2009 International Conference on Extended Data Base Technology (EDBT 2009), pp. 815–826 (2009)
5. Li, G., Feng, J., Wang, J., Yu, B., He, Y.: Race: Finding and Ranking Compact Connected Trees for Keyword Proximity Search over XML Documents. In: WWW, pp. 1045–1046 (2008)
6. Li, G., Feng, J., Wang, J., Zhou, L.: Effective Keyword Search for Valuable LCAs over XML Documents. In: CIKM, pp. 31–40 (2007)
7. Li, Y., Yu, C., Jagadish, H.: Schema-Free XQuery. In: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB 2004), pp. 72–83 (2004)
8. Liu, Z., Chen, Y.: Identifying Meaningful Return Information for XML Keyword Search. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD 2007), pp. 329–340 (2007)
9. Liu, Z., Walker, J., Chen, Y.: XSeek: A Semantic XML Search Engine Using Keywords. In: Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB 2007), pp. 1330–1333 (2007)
10. Sun, C., Chan, C., Goenka, A.: Multiway SLCA-based Keyword Search in XML Data. In: WWW, pp. 1043–1052 (2007)
11. Xu, Y., Papakonstantinou, Y.: Efficient Keyword Search for Smallest LCAs in XML Databases. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD 2005), pp. 537–538 (2005)
12. Xu, Y., Papakonstantinou, Y.: Efficient LCA Based Keyword Search in XML Data. In: Proc. 2008 International Conference on Extended Data Base Technology (EDBT 2008), pp. 535–546 (2008)
13. Yang, W., Zhu, H.: Semantic-Distance Based Clustering for XML Keyword Search. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010. LNCS, vol. 6119, pp. 398–409. Springer, Heidelberg (2010)
14. Zhou, R., Liu, C., Li, J.: Fast ELCA computation for keyword queries on XML data. In: Proc. 2010 International Conference on Extended Data Base Technology (EDBT 2010), pp. 549–560 (2010)

Steering Time-Dependent Estimation of Posteriors with Hyperparameter Indexing in Bayesian Topic Models

Tomonari Masada[1], Atsuhiro Takasu[2], Yuichiro Shibata[1], and Kiyoshi Oguri[1]

[1] Nagasaki University, 1-14 Bunkyo-machi, Nagasaki-shi, Nagasaki, Japan
{masada,shibata,oguri}@nagasaki-u.ac.jp
[2] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
takasu@nii.ac.jp

Abstract. This paper provides a new approach to topical trend analysis. Our aim is to improve the generalization power of latent Dirichlet allocation (LDA) by using document timestamps. Many previous works model topical trends by making latent topic distributions time-dependent. We propose a straightforward approach by preparing a different word multinomial distribution for each time point. Since this approach increases the number of parameters, overfitting becomes a critical issue. Our contribution to this issue is two-fold. First, we propose an effective way of defining Dirichlet priors over the word multinomials. Second, we propose a special scheduling of variational Bayesian (VB) inference. Comprehensive experiments with six datasets prove that our approach can improve LDA and also Topics over Time, a well-known variant of LDA, in terms of test data perplexity in the framework of VB inference.

Keywords: Bayesian methods, topic models, trend analysis, variational inference, parallelization.

1 Introduction

This paper provides a simple and efficient approach to Bayesian analysis of topic time-dependency for large-scale document sets. Our aim is to improve the generalization power of latent Dirichlet allocation (LDA) [3] in terms of test data perplexity by proposing a topic model where we use document timestamps as a key factor for improvement. Our approach is based on the intuition that a careful analysis of word frequency differences among time points will help topic extraction by LDA-like Bayesian models. We propose a simple time-dependent variant of LDA and devise a special scheduling of variational Bayesian (VB) inference [3]. In our model, a different word multinomial distribution is prepared for each time point. When we have T time points and K topics, $T \times K$ word multinomials are prepared in total. As this leads to a large number of parameters, overfitting becomes critical. Our contribution to this issue is two-fold:

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 435–447, 2011.

1. We propose a non-trivial way of defining Dirichlet priors over $T \times K$ word multinomials. When we define a single common Dirichlet prior over all these word multinomials, overfitting is too strongly suppressed. Thus we propose an effective way of defining Dirichlet priors over these word multinomials.
2. We propose a special scheduling of VB inference. As an initialization, VB for LDA described in [3] is conducted for a certain number of iterations, and our model is initialized with the estimated parameters. We further introduce another twist as a finalization whose details will be exposed later.

We conduct comprehensive experiments on six datasets, four of which are in English, one in Japanese, and the rest one in Korean. The largest dataset contains 368,000 documents, a set of one-year news articles, and 32,800,000 unique document-word pairs. Since our model has a simple construction, it does not sacrifice the efficiency in computation cost for mathematical sophistication and thus can easily handle large datasets. Our approach requires at most 1.5 times as much computation time as LDA in the framework of VB inference parallelized for multi-core CPU. With this efficiency, our approach can improve the generalization power of LDA and further that of Topics over Time (TOT) [16], a well-known time-dependent variant of LDA, by up to 15 percent.

The rest of the paper is organized as follows. Section 2 discusses previous works related to topical trend analysis. Section 3 provides the details of our proposal. Section 4 presents the settings and the results of our evaluation experiments. Section 5 concludes the paper with discussions and future work.

2 Previous Works

Recent Web analysis has focused on processing *timestamped* documents, because we can observe an immense increase in the number of realtime posts sent to a variety of SNS sites, e.g. Twitter, Facebook, etc. Bayesian approach, one of the mainstreams in text mining, also seeks a method for analyzing time-dependency of topics latent in large-scale document sets to capture salient topical trends.

We briefly introduce LDA [3], a standard document model in Bayesian text mining. With LDA, we can take each document as a conglomerate of multiple semantic contents. LDA characterizes each document by a probability distribution defined over a fixed number K of latent topics. Precisely, LDA attaches a multinomial distribution $\texttt{Multi}(\theta_j)$ defined over the topics $\{c_1, \ldots, c_K\}$ to each of the given documents $\{d_1, \ldots, d_J\}$, where θ_j denotes the parameters $(\theta_{j1}, \ldots, \theta_{jK})$ of the topic multinomial for document d_j. We can regard θ_{jk} as the probability that any word token in document d_j expresses topic c_k, not the other topics. Further, LDA characterizes each topic by the multinomial $\texttt{Multi}(\phi_k)$ defined over the fixed word set $\{v_1, \ldots, v_W\}$, where ϕ_k denotes the parameters $(\phi_{k1}, \ldots, \phi_{kW})$ of the word multinomial for topic c_k. We can regard ϕ_{kw} as the probability that topic c_k is expressed by any token of word v_w, not of the other words.

A remarkable feature of LDA is that all topic multinomial parameters $\{\theta_1, \ldots, \theta_J\}$ are drawn from a single common Dirichlet prior distribution $\texttt{Di}(\alpha)$, where α denotes the set of the K hyperparamters $(\alpha_1, \ldots, \alpha_K)$. Further, all word

multinomial parameters $\{\phi_1, \ldots, \phi_K\}$ are drawn from another single common Dirichlet prior $\mathtt{Di}(\beta)$, where β denotes the W hyperparamters $(\beta_1, \ldots, \beta_W)$. In the following, we simply call $\mathtt{Di}(\alpha)$ topic Dirichlet prior and call $\mathtt{Di}(\beta)$ word Dirichlet prior. With these two priors, LDA can reduce the diversity among the topic multinomials and also the diversity among the word multinomials, and thus can achieve a generalization power better than PLSI [7].

When making LDA time-dependent, we can consider the following two assumptions [12]: (i) Topic distributions vary along time; (ii) Word distributions vary along time. Many previous works [2,10,11,15] adopt the former assumption. While Srebro et al. [12] give discussions supporting the former, other works adopt the latter [8] or combine both [9]. In this paper, we adopt the latter assumption and prepare a different word multinomial distribution for each time point. In [11], a highly sophisticated modeling is devised, and the former assumption is combined with the assumption of infinite topics [13]. While our approach may be combined with nonparametric approach, we do not pursue this direction in this paper. The topic model in [8] is similar to ours, because this model has a different word multinomial for each time point. However, this work further considers multiscale effects of the past word frequencies at each time point and realizes a flexible time-dependent posterior estimation. Consequently, the model becomes quite complicated and requires an approximated inference, whose implementation becomes intricate when we attempt to achieve a tolerable computation cost. In contrast, our approach seeks an efficient balance between generalization power and computation cost.

3 Method

3.1 Model Construction

In this paper, we model the time-dependency of topics by using a different word multinomial for each of the given time points $\{s_1, \ldots, s_T\}$. Therefore, our model has $T \times K$ word multinomials $\mathtt{Multi}(\phi_{tk})$, $t = 1, \ldots, T$, $k = 1, \ldots, K$, where ϕ_{tk} denotes the multinomial parameters $(\phi_{tk1}, \ldots, \phi_{tkW})$. That is, we have $T \times K \times W$ word multinomial parameters in total, as in [8]. However, we take an approach different from [8] in defining Dirichlet priors over the word multinomials.

As is discussed in Section 2, LDA has K word multinomials each corresponding to a different topic and defines a single Dirichlet prior over these K multinomials. Therefore, we first defined a single Dirichlet prior over all $T \times K$ multinomials in our model. However, a preliminary experiment showed that overfitting was too strongly suppressed and that a poor generalization power was obtained.

Therefore, in another preliminary experiment, we tested the three options in Table 1. As a result, Option 1 achieved success for many datasets. Option 1 defines a common Dirichlet prior $\mathtt{Di}(\beta_k)$ over the T word multinomials $\mathtt{Multi}(\phi_{1k}), \ldots, \mathtt{Multi}(\phi_{Tk})$ for each k. β_k refers to the hyperparameters $(\beta_{k1}, \ldots, \beta_{kW})$ of $\mathtt{Di}(\beta_k)$ prepared for topic c_k. As the T word multinomials $\mathtt{Multi}(\phi_{1k}), \ldots, \mathtt{Multi}(\phi_{Tk})$ are drawn from the same prior, we have a smoothing only separately for each topic, not over all word multinomials. Therefore, we

Table 1. Three options for defining word Dirichlet priors

Option 1	Prepare a different prior $\texttt{Di}(\beta_k)$ for each of the K disjoint groups of word multinomials, where each group corresponds to a different topic c_k and contains T word multinomials $\texttt{Multi}(\phi_{1k}), \ldots, \texttt{Multi}(\phi_{Tk})$. This option gives $K \times W$ word Dirichlet hyperparamters in total.
Option 2	Prepare a different prior $\texttt{Di}(\beta_t)$ for each of the T disjoint groups of word multinomials, where each group corresponds to a different time point s_t and contains K multinomials $\texttt{Multi}(\phi_{t1}), \ldots, \texttt{Multi}(\phi_{tK})$. This option gives $T \times W$ word Dirichlet hyperparamters in total.
Option 3	Prepare a different prior $\texttt{Di}(\beta_{tk})$ for each of the $T \times K$ word multinomials separately. This option gives $T \times K \times W$ word Dirichlet hyperparamters in total.

can achieve a more moderate smoothing than a single common prior. However, Option 3 also gave impressive results for some datasets. When we take Option 3, the hyperparameters of the word Dirichlet priors are endowed with fully fine-grained indices β_{tkw}. Since these indices are as fine-grained as the indices of the word multinomial parameters ϕ_{tkw}, Option 3 gives a smoothing effect more moderate than Option 1 and thus showed poor results due to overfitting for several datasets. However, since Option 3 was effective for some datasets, we combine Option 1 with Option 3 in our inference, as will be described in Section 3.2. The detailed results of the preliminary experiments will be given in Section 4.4.

The topic model proposed in [8] adopts Option 3 and does not consider Option 1 and Option 2. That is, the hyperparameters of the word Dirichlet priors are endowed with fully fine-grained indices β_{tkw}. The model in [8] seems to avoid overfitting with multiscale analysis, which can exploit the interaction among word Dirichlet priors attached to different time points. This may cause a smoothing effect and thus may lead to a good generalization power. In contrast, we avoid such complication in modeling and consider various ways of indexing the hyperparameters of the word Dirichlet priors, as in Table 1, to obtain a special scheduling of VB inference where some of these options are combined.

3.2 Posterior Inference

For posterior inference, we adopt variational Bayesian (VB) inference [3]. One reason of this choice is that parallelization is easier than collapsed Gibbs sampling (CGS) [5] and collapsed variational Bayesian (CVB) inference [14]. Many operations in VB inference are embarrassingly parallel like EM alogorithm [4], and thus our approach can scale up to larger datasets. Another reason is that VB achieves a generalization power comparable with CGS and CVB [1].

Due to space limitation, we only give an outline of the formula derivation for our VB, which is similar to that for LDA [3]. Let $(\iota_{j1}, \ldots, \iota_{jK})$ be the parameters of the variational Dirichlet posterior defined over the topics $\{c_1, \ldots, c_K\}$ and attached to document d_j. Intuitively, ι_{jk} tells how strongly the word tokens in document d_j express topic c_k. Further, let $(\zeta_{tk1}, \ldots, \zeta_{tkW})$ be the parameters

of the variational Dirichlet posterior defined over the words $\{v_1, \ldots, v_W\}$ and attached to the pair of time point s_t and topic c_k. Intuitively, ζ_{tkw} tells how strongly topic c_k is expressed by the tokens of word v_w at time point s_t. With a variational approximation, we obtain the lower bound $\mathcal{L}$ of the log evidence as:

$$\begin{aligned}\mathcal{L} = &\sum_{j,w,k} n_{jw}\pi_{jwk}\Big\{\Psi(\iota_{jk}) - \Psi(\sum_k \iota_{jk}) + \Psi(\zeta_{t_jkw}) - \Psi(\sum_w \zeta_{t_jkw}) - \log \pi_{jwk}\Big\} \\ &- \sum_{j,k}\Big[\Gamma(\alpha_k) - \Gamma(\iota_{jk}) - (\alpha_k - \iota_{jk})\Big\{\Psi(\iota_{jk}) - \Psi(\sum_k \iota_{jk})\Big\}\Big] \\ &- \sum_{t,k,w}\Big[\Gamma(\beta_{kw}) - \Gamma(\zeta_{tkw}) - (\beta_{tw} - \zeta_{tkw})\Big\{\Psi(\zeta_{t_jkw}) - \Psi(\sum_w \zeta_{t_jkw})\Big\}\Big] \ , \quad (1)\end{aligned}$$

where π_{jwk}, satisfying $\sum_k \pi_{jwk} = 1$, refers to the approximated posterior probability that a token of word v_w in document d_j expresses topic c_k. n_{jw} denotes the number of the tokens of word v_w in document d_j. Further, Γ (resp. Ψ) denotes the gamma (resp. digamma) function, and $t_j \in \{1, \ldots, T\}$ is the index of the timestamp of document d_j.

From the partial derivatives of $\mathcal{L}$, we obtain the following update formulas:

$$\pi_{jwk} \propto \exp\Big\{\Psi(\iota_{jk}) - \Psi(\sum_k \iota_{jk}) + \Psi(\zeta_{t_jkw}) - \Psi(\sum_w \zeta_{t_jkw})\Big\} \ , \quad (2)$$

$$\iota_{jk} = \alpha_k + \sum_w n_{jw}\pi_{jwk} \ , \quad (3)$$

$$\zeta_{tkw} = \beta_{kw} + \sum_{\{j:t_j=t\}} n_{jw}\pi_{jwk} \ , \quad (4)$$

$$\alpha_k = \Psi^{-1}\Big(\sum_k \alpha_k + \sum_j \{\Psi(\iota_{jk}) - \Psi(\sum_k \iota_{jk})\}/J\Big) \ , \quad (5)$$

$$\beta_{kw} = \Psi^{-1}\Big(\sum_w \beta_{kw} + \sum_t \{\Psi(\zeta_{tkw}) - \Psi(\sum_w \zeta_{tkw})\}/T\Big) \ , \quad (6)$$

where Ψ^{-1} is the inverse of the digamma function.

While we used the formulas above in a preliminary experiment, any start from a random initialization was likely to find a poor local optimum. Therefore, we propose a special *initialization*. We first conduct VB for LDA with a random initialization by ignoring all timestamps. After a fixed number of iterations (50 iterations in our experiments) of this VB inference, we initialize the parameters of our model with the parameters estimated by this VB for LDA and start the VB for our model by using the update formulas shown above.

In the VB for LDA as an initialization, we use the following update formulas:

$$\pi_{jwk} \propto \exp\Big\{\Psi(\iota_{jk}) - \Psi(\sum_k \iota_{jk}) + \Psi(\eta_{kw}) - \Psi(\sum_w \eta_{kw})\Big\} \ , \quad (7)$$

$$\iota_{jk} = \alpha_k + \sum_w n_{jw}\pi_{jwk} \ , \quad (8)$$

$$\eta_{kw} = \beta_w + \sum_j n_{jw}\pi_{jwk} \ , \tag{9}$$

where η_{kw} is the approximated posterior parameter telling how strongly topic c_k is expressed by the tokens of word v_w. These three formulas correspond to Eqs. (2), (3), and (4), respectively. In our special scheduling, we first conduct the VB inference for LDA by using Eqs. (7), (8), and (9) and then initialize the parameters π_{jwk} and ι_{jk} of our model with the estimations obtained by this VB for LDA. At the same time, we initialize the posterior parameters ζ_{tkw}, $k = 1, \ldots, K$, $w = 1, \ldots, W$ of our model as $\zeta_{tkw} = \eta_{kw}$ for each t.

As is discussed in Section 3.1, Option 3 in Table 1 sometimes gave impressive results in the preliminary experiment. Therefore, we use Option 3 as a *finalization* of our VB inference. After conducting VB for LDA as an initialization, we train our model with Option 1 for a large enough number of iterations (140 iterations in our experiments). Then, we use the hyperparameters β_{kw} to initialize the hyperparameters β_{tkw} in Option 3. Precisely, we set $\beta_{tkw} = \beta_{kw}$ for each t. After this, we conduct a small number of iterations of VB inference update (10 iterations in our experiments) with Option 3 as a finalization. The update formula for β_{tkw} in Option 3 can be written as follows:

$$\beta_{tkw} = \Psi^{-1}\Big(\sum_w \beta_{tkw} + \Psi(\zeta_{tkw}) - \Psi(\sum_w \zeta_{tkw})\Big) \ . \tag{10}$$

With respect to the effectiveness of Option 3, Section 4.4 includes detailed discussions based on the experimental results.

Our three-stage VB inference scheduling is summarized in the left panel of Figure 1. While the number of iterations at each stage is determined based on the preliminary experiments, the important point is that we give a *gradually increasing degree of freedom* to word posterior estimation as VB inference

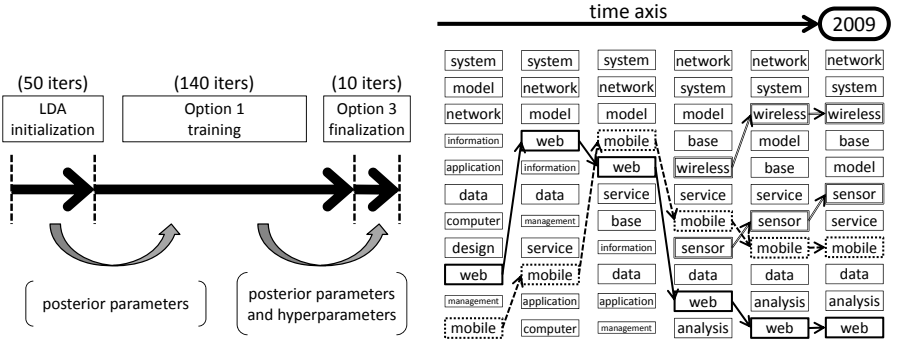


Fig. 1. The proposed inference scheduling for our model (left panel) and an example of the topical trends extracted by our approach (right panel). Each column in the right panel corresponds to a different time point (year) and includes the words sorted in the decreasing order of ζ_{tkw} (cf. Eq. (4)) from top to bottom.

proceeds. In this manner, we *S*teer *T*ime-dependent *E*stimation of *P*osteriors with *HY*perparameter indexing. We call our approach *STEPHY* by concatenating the italicized uppercase letters in the previous sentence.

4 Experiments

4.1 Datasets

We prepared the six datasets in Table 2 for our experiments. J, W, T, and P in Table 2 are the numbers of documents, different words, different time points, and different document-word pairs, respectively. NIPS is the dataset often used in the experiments for machine learning. We used a well-cleaned version appearing in [6][1]. We regarded each publication year as a different time point. This dataset is far smaller than the other five datasets. DBLP dataset is a part of the XML data available at the DBLP Web site[2]. We regarded the paper title as a document and the publication year as a timestamp. Since we used the papers dated from 1990 to 2009, T is equal to 20. DONGA is a set of Korean news articles issued in 2008 and were downloaded from the politics section of the Donga Ilbo Web site[3]. Each article was processed by KLT morphological analyzer[4] to segment each sentence into word tokens. We regarded each week as a single time point. Consequently, we have 53 different time points. TDT is the dataset prepared for the 2002-2003 Evaluation in Topic Detection and Tracking, Phase 4[5]. We regarded the date of each document as a timestamp. This dataset has the largest number of different time points among the six datasets. NSF is the dataset available at UCI machine learning repository[6]. Each document has an ID (e.g. "a9000006"). We regard its first two digits (e.g. "90") as a timestamp. The resulting timestamps range from 90 (i.e., 1990) to 02 (i.e., 2002). YOMI is a set of the Japanese news articles of Yomiuri newspaper published in 2005[7]. The documents were processed by MeCab[8] morphological analyzer to extract words. As in case of DONGA, we regarded each week as a timestamp. For all datasets, we removed the words of low and high frequency by following a common practice of text mining.

4.2 Settings

For comparison, we also adopt VB inference for both LDA and TOT. The VB for LDA is explained in [3]. For TOT, no preceding works report the update formulas of VB. Since the formulas are a slight modification of those of LDA,

[1] http://www.cs.huji.ac.il/~amitg/htmm.html
[2] http://dblp.uni-trier.de/xml/dblp.xml.gz
[3] http://news.donga.com/Politics
[4] http://nlp.kookmin.ac.kr/HAM/kor/
[5] http://projects.ldc.upenn.edu/TDT4/
[6] http://archive.ics.uci.edu/ml/
[7] http://www.ndk.co.jp/yomiuri/
[8] http://mecab.sourceforge.net/

Table 2. Specifications of six datasets used in our experiments

	J	W	T	P		J	W	T	P
NIPS	1,740	11,998	13	919,916	TDT	96,256	51,849	123	11,460,231
DBLP	1,235,988	273,173	20	7,814,175	NSF	128,181	25,325	13	10,388,976
DONGA	24,093	71,621	53	7,949,288	YOMI	367,910	84,060	52	32,762,456

we omit the details here. We only note that the parameters of per-topic Beta distributions in TOT should be rescaled as in case of CGS [16]. We set the rescaling factor to 0.7 based on preliminary trials. Also for TOT, any start from a random initialization gave a poor generalization power. Therefore, we first train LDA and use the resulting posteriors for initializing TOT as in STEPHY.

The experiments are conducted on a Fedora 12 Linux PC equipped with Intel Core i7 920 CPU at 2.67 GHz and 12 Gbytes of main memory. For all cases in our experiments, this main memory size is enough to store all of the input data and the model parameters. To exploit the full potential of our multi-core CPU, we parallelize the operations in VB with OpenMP library by implementing the inference from scratch. Every execution time reported in this paper is a wall-clock time obtained by running eight threads on the four cores of our CPU.

4.3 Evaluation Measure

We evaluate the generalization power of each compared approach by test data perplexity, which tells how well each topic model can generalize to test data. We randomly select 10 percent word tokens from the entire dataset as test word tokens and use them for calculating the perplexity defined as follows:

$$perplexity \equiv \exp\left\{ -\sum_{j}\sum_{i} \log \sum_{k} \bar{\iota}_{jk}\bar{\zeta}_{t_j k x_{ji}} \Big/ N_{test} \right\} , \tag{11}$$

where N_{test} is the number of the test word tokens, $t_j \in \{1, \ldots, T\}$ is the index of the timestamp of document d_j, and $x_{ji} \in \{1, \ldots, W\}$ is the index of the word appearing as the ith test word token of document d_j. The summation $\sum_i$ in Eq. (11) is taken only over the test word tokens. Further, $\bar{\iota}_{jk}$ and $\bar{\zeta}_{tkw}$ are the posterior probabilities obtained by normalizing the posterior Dirichlet parameters ι_{jk} and ζ_{tkw}, appearing in Eq. (3) and Eq. (4), as follows:

$$\bar{\iota}_{jk} = \frac{\iota_{jk}}{\sum_k \iota_{jk}} , \quad \bar{\zeta}_{tkw} = \frac{\zeta_{tkw}}{\sum_{w'} \zeta_{tkw'}} . \tag{12}$$

A smaller perplexity corresponds to a better generalization power. The perplexity for LDA and TOT is defined similarly by using η_{kw} in Eq. (9) instead of ζ_{tkw}.

4.4 Preliminary Experiments

Before giving the results of the main experiment comparing STEPHY with LDA and TOT, we overview the results of our preliminary experiments in Table 3 and

Table 3. Results of the preliminary experiment comparing initialization methods

	Option 1 (Random init.)	Option 1 (LDA init.)
NIPS	1604.0±9.8	**1394.4±10.2**
DBLP	3299.3±9.4	**3065.9±12.2**
DONGA	3026.2±79.8	**2195.9±23.1**
TDT	3963.7±71.4	**2049.6±15.3**
NSF	3382.5±6.3	**1700.5±13.6**
YOMI	3477.7±76.4	**2768.2±20.1**

Table 4. Results of the preliminary experiment comparing various options for hyperparameter indexing

	Option 1	Option 2	Option 3	Single common prior
NIPS	1394.4±10.2	1633.9±8.9	1779.0±10.7	**1334.0±11.5**
DBLP	**3065.9±12.2**	3625.3±9.1	3327.5±21.5	3311.8±17.7
DONGA	**2195.9±23.1**	3211.4±13.4	2310.6±19.0	2599.2±19.5
TDT	**2049.6±15.3**	3811.4±12.0	2706.8±15.0	2787.6±41.4
NSF	**1700.5±13.6**	3422.7±4.3	1723.2±12.8	3373.2±4.9
YOMI	**2768.2±20.1**	4847.2±24.6	3130.8±29.8	3041.6±34.0

Table 4 to support the discussions in Section 3.1 and Section 3.2. These tables give the test data perplexity at the 200th iteration, i.e., the final iteration, of the VB inference when we set $K = 50$. Each perplexity is averaged over 20 different execution instances, and the corresponding standard deviation is also presented.

Table 3 shows the effect of our special initialization. The leftmost column includes the tags of the six datasets. When we train our model with Option 1 in Table 1 after a random initialization, we obtain the test data perplexity shown in the center column. When we train our model with Option 1 after an initialization using the posterior estimation of LDA, we obtain the perplexity in the rightmost column. Table 3 proves that the initialization with VB for LDA gives a better generalization power than the random initialization for all datasets.

Table 4 shows a comparison between the various ways of indexing the hyperparameters of word Dirichlet priors. We applied the initialization with VB for LDA to each compared case. When we prepare a single common word Dirichlet prior for all $T \times K$ word multinomials, we obtain the perplexity in the rightmost column. When we adopt Option 1, Option 2, and Option 3 in Table 1, we obtain the results in the second, third, and fourth column, respectively.

When we only define a single common word Dirichlet prior, the test data perplexity is poor for many datasets, as is shown in the rightmost column of Table 4. While the perplexity for NIPS dataset is occasionally good, this dataset is far smaller than the other datasets and does not represent the general situation. It seems that a single word Dirichlet prior is enough to cover the topical diversity latent in NIPS dataset. In contrast, the perplexity for NSF dataset is of disastrous level. This may be because overfitting is too strongly suppressed by a single common prior. Option 2 also gives a poor perplexity for many datasets. Option

Table 5. Test data perplexity and wall-clock computation time after 200 iterations

	Test data perplexity			Computation time (in sec.)		
	STEPHY	TOT	LDA	STEPHY	TOT	LDA
NIPS	**1407.9±13.8**	1685.2±9.8	1659.9±8.4	489.0 (×1.46)	373.5	334.7
DBLP	**3027.6±17.3**	3439.4±39.6	3446.2±30.3	4737.8 (×1.39)	3700.6	3411.4
DONGA	**2062.2±26.7**	2524.4±25.7	2475.1±24.9	3925.5 (×1.39)	3130.1	2829.2
TDT	**1897.1±31.7**	2005.5±11.6	1988.7±10.7	5354.4 (×1.39)	4292.3	3842.8
NSF	1684.1±18.8	1689.5±12.4	1691.8±14.2	3800.4 (×1.09)	3876.6	3473.8
YOMI	**2671.8±33.3**	2850.2±18.0	2844.2±14.9	13380.5 (×1.18)	12701.5	11390.8

2 defines a common word Dirichlet prior $\texttt{Di}(\beta_t)$ over the K word multinomials $\texttt{Multi}(\phi_{t1}), \ldots, \texttt{Multi}(\phi_{tK})$ each corresponding to a different topic. This definition is used for each time point s_t separately. Therefore, while Option 2 can differentiate between various time points, the topical diversity is not well captured, because the word posteriors corresponding to different topics share the same Dirichlet prior. In contrast, the perplexity achieved by Option 3 is fairly good. Option 3 gives the second best perplexity for DONGA, TDT, and NSF datasets. However, Option 3 requires a large computation time, because Option 3 gives $T \times K \times W$ word Dirichlet hyperparameters in total and thus requires considerable time for the hyperparameter update using Eq. (10). Therefore, we adopt Option 1, giving the best result for all datasets except NIPS, as the main driving force and use Option 3 for finalization. Based on this line of reasoning, we propose an inference scheduling drawn in the left panel of Figure 1.

4.5 Main Experiment

The results of our main experiment are summarized in Table 5. Both test data perplexity and computation time are obtained at the 200th iteration, i.e., at the final iteration, and are averaged over the results of 20 different execution instances. We also include the corresponding standard deviation for test data perplexity. Table 5 shows that STEPHY gives a smaller perplexity than LDA and TOT for all datasets. Especially, for DONGA dataset, the perplexity is reduced by 16.7 percent when compared with LDA. While the margin of improvement is not significantly large only for NSF dataset, we can say that STEPHY can put an improvement into the VB inference framework of LDA-like topic models.

Further, we can compare STEPHY in Table 5 with Option 1 in Table 4. As the left panel of Figure 1 shows, STEPHY conducts 10 iterations with Option 3 as a finalization after the 140 iterations with Option 1. By comparing STEPHY in Table 5 with Option 1 in Table 4, it can be observed that this finalization improves the perplexity for the following five datasets: DBLP ($3027.6 \pm 17.3 < 3065.9 \pm 12.2$), DONGA ($2062.2 \pm 26.7 < 2195.9 \pm 23.1$), TDT ($1897.1 \pm 31.7 < 2049.6 \pm 15.3$), NSF ($1684.1 \pm 18.8 < 1700.5 \pm 13.6$), and YOMI ($2671.8 \pm 33.3 < 2768.2 \pm 20.1$). We can contend that the finalization with Option 3 works.

The comparison experiment also shows that the increase in computation cost brought by our approach is moderate. Table 5 includes the wall-clock

computation time each compared method requires for each dataset. The computation time of STEPHY is at most 1.46 times of LDA. With this increase in computation time, we can achieve a significant improvement shown in Table 5. While TOT requires less running time than STEPHY, TOT improves LDA only for DBLP and NSF datasets with a small margin. We can say that STEPHY provides a good balancing between generalization power and computation cost. We additionally conducted a set of experiments also for the case $K = 100$, i.e., the case where the number of topics is 100. The results, omitted due to space limitation, confirm our conclusion on the efficiency of STEPHY.

The right panel of Figure 1 gives an example of the topical trend extracted by STEPHY from DBLP dataset. Each column corresponds to a different time point and includes the words sorted in the decreasing order of the posterior parameters ζ_{tkw} from top to bottom for one topic arbitrarily selected from the 50 topics. We can interpret the parameter ζ_{tkw} as showing how popularly word v_w is used to express topic c_k at time point s_t. In the right panel of Figure 1, two or three top-ranked words keep their positions over many different time points. However, some explicit topical trends can be observed under these top-ranked words. For example, the word "web" shows a peak around five or six years ago from 2009, and the word "mobile" shows a stable popularity in recent three or four years. Further, the rapid growth of the popularity of the words "wireless" and "sensor" may correspond to the recent rise of the trend related to wireless and sensor networks. Since our approach provides a different word posterior distribution for each time point, this type of trend analysis can be easily conducted only by inspecting the estimated values of ζ_{tkw} along the time axis.

5 Conclusion

In this paper, we propose a simple time-dependent variant of LDA and an effective VB for the proposed model. STEPHY, our total schema for time-dependent topic modeling, improves LDA and TOT in terms of test data perplexity and only increases the computation time of LDA by at most a factor of 1.5.

With respect to the balancing between generalization power and computation cost, we can add the following discussion. STEPHY improves LDA only based on the *intra-epoch* document similarity assumption, i.e., the assumption that the documents having the same timestamp are semantically related to each other. We do not explicitly model any interrelationships of word frequencies over neighboring time points. Therefore, our model does not require an intricate inference. In contrast, many time-dependent topic models are further based on the *inter-epoch* similarity assumption, i.e., the assumption that the documents having different but close timestamps are also semantically related, and intensively exploit the topical dependency over neighboring time points [2,8,9,10,11,15,16]. Consequently, the inference requires detailed tricks and becomes less scalable. Our experiments show that, with the intra-epoch similarity, STEPHY achieves an efficient balance between computation cost and generalization power.

STEPHY leaves intact the model construction related to latent topics in LDA. Therefore, one possible future work is to combine our approach with the assumption of infinite topics [13,11], though we should shift the balance against computational efficiency and check if STEPHY can contribute more than the nonparametric approach that takes advantage of intricate inference.

Another more challenging future work is to apply STEPHY to the probabilistic models where each topic is characterized by a probability distribution other than multinomial distribution. By steering the time-dependent estimation of posteriors with some hyperparameter indexing strategies like those given in Table 1, we can make a similar proposal for efficiently exploring the parameter space also with respect to those models.

Acknowledgement

This work was supported in part by the Nagasaki University Strategy for Fostering Young Scientists with funding provided by the Special Coordination Funds for Promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology (MEXT).

References

1. Asuncion, A., Welling, M., Smyth, P., Teh, Y.-W.: On smoothing and inference for topic models. In: Proc. of UAI 2009 (2009)
2. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: Proceedings of ICML 2006, pp. 113–120 (2006)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
4. Chu, C.-T., Kim, S.-K., Lin, Y.-A., Yu, Y.-Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In: Proceedings of NIPS 2006, pp. 281–288 (2006)
5. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences of the United States of America 101(1), 5228–5235 (2004)
6. Gruber, A., Rosen-Zvi, M., Weiss, Y.: Hidden topic Markov models. In: Proceedings of AISTATS 2007 (2007)
7. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of SIGIR 1999, pp. 50–57 (1999)
8. Iwata, T., Yamada, T., Sakurai, Y., Ueda, N.: Online multiscale dynamic topic models. In: Proceedings of KDD 2010, pp. 663–672 (2010)
9. Nallapati, R.M., Ditmore, S., Lafferty, J.D., Ung, K.: Multiscale topic tomography. In: Proceedings of KDD 2007, pp. 520–529 (2007)
10. Pruteanu-Malinici, I., Ren, L., Paisley, J., Wang, E., Carin, L.: Hierarchical Bayesian modeling of topics in time-stamped documents. IEEE Trans. Pattern Anal. Mach. Intell. 32(6), 996–1011 (2010)
11. Ren, L., Dunson, D.B., Carin, L.: The dynamic hierarchical Dirichlet process. In: Proceedings of ICML 2008, pp. 824–831 (2008)
12. Srebro, N., Roweis, S.: Time-varying topic models using dependent Dirichlet processes. Technical report, Dept. of Computer Science, Univ. of Toronto (2005)

13. Teh, Y.-W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical Dirichlet processes. Journal of the American Statistical Association 101(476), 1566–1581 (2006)
14. Teh, Y.-W., Newman, D., Welling, M.: A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In: Proceedings of NIPS 2006, pp. 1353–1360 (2006)
15. Wang, C., Blei, D., Heckerman, D.: Continuous time dynamic topic models. In: Proceedings of UAI 2008, pp. 579–586 (2008)
16. Wang, X.-R., McCallum, A.: Topics over time: A non-Markov continuous-time model of topical trends. In: Proceedings of KDD 2006, pp. 424–433 (2006)

Constrained LDA for Grouping Product Features in Opinion Mining

Zhongwu Zhai[1], Bing Liu[2], Hua Xu[1], and Peifa Jia[1]

[1] State Key Lab of Intelligent Tech. & Sys., State Key Lab of Intelligent Tech. & Sys., Dept. of Comp. Sci. & Tech., Tsinghua Univ.
zhaizhongwu@gmail.com
[2] Dept. of Comp. Sci., University of Illinois at Chicago
liub@cs.uic.edu

Abstract. In opinion mining of product reviews, one often wants to produce a summary of opinions based on product features. However, for the same feature, people can express it with different words and phrases. To produce an effective summary, these words and phrases, which are domain synonyms, need to be grouped under the same feature. Topic modeling is a suitable method for the task. However, instead of simply letting topic modeling find groupings freely, we believe it is possible to do better by giving it some pre-existing knowledge in the form of automatically extracted constraints. In this paper, we first extend a popular topic modeling method, called Latent Dirichlet Allocation (LDA), with the ability to process *large* scale constraints. Then, two novel methods are proposed to extract two types of constraints automatically. Finally, the resulting *constrained-LDA* and the extracted constraints are applied to group product features. Experiments show that *constrained-LDA* outperforms the original LDA and the latest *mLSA* by a large margin.

Keywords: Opinion Mining, Feature Grouping, Constrained LDA.

1 Introduction

One form of opinion mining in product reviews is to produce a feature-based summary [1]. In this model, product features are first identified, and positive and negative opinions on them are aggregated to produce a summary of opinions on the features. Product features are attributes and components of products, e.g., "picture quality", "battery life" and "zoom" of a digital camera.

In reviews (or any writings), people often use different words and phrases to describe the same product feature. For example, "picture" and "photo" refer to the same feature for cameras. Grouping such synonyms is critical for opinion summary. Although WorldNet and other thesaurus dictionaries can help to some extent, they are insufficient because many synonyms are domain dependent. For example, "movie" and "picture" are synonyms in movie reviews, but they are not synonyms in camera reviews as "picture" is more likely to be synonymous to "photo" while "movie" to "video". This paper deals with this problem, i.e.,

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 448–459, 2011.

grouping domain synonym features. We assume that all the feature expressions have been identified by an existing algorithm.

Topic modeling is a principled approach to solving this problem as it groups terms of the same topic into one group. This paper takes this approach. However, we believe instead of letting a topic modeling method to run completely unsupervised, some pre-existing knowledge can be incorporated into the process to produce better results. The pre-existing knowledge can be inputted manually or extracted automatically. In this work, we extract such knowledge automatically.

Topic modeling methods can be seen as clustering algorithms that cluster terms into homogeneous topics (or clusters). In the classic clustering research in data mining, there is a class of semi-supervised clustering algorithms which allow constraints to be set as prior knowledge to restrict or to guide clustering algorithms to produce more meaningful clusters to human users [2, 3]. These constraints are in the forms of must-links and cannot-links. A must-link constraint specifies that two data instances *must* be in the same cluster. A cannot-link constraint specifies that two data instances *cannot* be in the same cluster.

In this paper, we incorporate these two types of constraints into the popular topic modeling method LDA to produce a semi-supervised LDA method, called *constrained-LDA*. To the best of our knowledge, this is the first constrained LDA model which can process large scale constraints in the forms of must-links and cannot-links. There are two existing work by Andrzejewski and Zhu [4, 5] that are related to the proposed model. However, [4] only considers must-link constraints. In [5], the number of maximal cliques grow exponentially in the process of encoding constraints. Thus, [5] cannot process a large number of constraints (see Section 2). As we will also see in the related work section, our method of incorporating the two types of constraints is entirely different from the way that they did.

Although we call them must-link and cannot-link constraints, they are treated as "soft" rather than "hard" constraints in the sense that they can be violated or relaxed in the topic modeling process. The relaxation mechanism is needed because some constraints may not be correct especially when the constraints are extracted automatically. In our case, all constraints are extracted automatically with no human involvement. Thus, the constraints may be more appropriately called probabilistic must-link and cannot-link constraints.

On extracting must-link and cannot-link constraints for our application, we use two observations. First, we observed that a review sentence may comment on several product features, e.g., "I like the picture quality, the battery life, and zoom of this camera" and "The picture quality is great, the battery life is also long, but the zoom is not good". From either of the sentences, we can see that the features, "picture quality", "battery life" and "zoom" are unlikely to be synonyms or belonging to the same topic simply because people normally will not repeat the same feature in the same sentence. This observation allows us to extract many cannot-link constraints automatically. As for must-links, we observed that two noun phrases that shared one or more words are likely to fall into the same topic, e.g., "battery life" and "battery power". Clearly, the

2 methods for identifying constraints are not perfect, i.e., they may find wrong constraints. The constraint relaxation mechanism comes to help.

Experiments show that the proposed constrained-LDA produces significantly better results than the original LDA and the latest multilevel topic modeling method, mLSA [6] which also uses LDA.

2 Related Work

This study is related to 2 research areas, topic modeling and synonym grouping.

Topic Modeling and LDA: Blei et al. [7] proposed the original LDA using EM estimation. Griffiths and Steyvers [8] applied Gibbs sampling to estimate LDA's parameters. Since these works, many variations have been proposed. In this paper, we only focus on the variations that add supervised information in the form of latent topic assignments.

Blei and McAuliffe [9] introduced a supervised latent Dirichlet allocation (sLDA). In sLDA, the authors added to LDA a response variable associated with each document, such as document's class label or document's rating. Ramage et al. [10] proposed a labeled LDA which considers the tag information of the documents. Chang and Blei [11] developed a relational topic model by adding the link information between documents. All these studies improve LDA by adding the labeled information of documents, whereas our constrained-LDA adds supervision to individual terms.

In [4], predefined topic-in-set knowledge (which means predefined terms for certain topics) was added to supervise the topic assignment for individual terms. Compared with our model, their model only used the must-link knowledge, not cannot-links. Moreover, our model's "topic-in-set knowledge" is updated dynamically after each Gibbs sampling, rather than fixed as predefined. Probability information is also introduced to the "topic-in-set knowledge".

In [5], must-link and cannot-link constraints were encoded with a Dirichlet Forest and were further incorporated into LDA. However, their model has a fatal limitation, as illustrated in Section 3.3 of [5], namely, the number of maximal cliques $Q^{(r)}$ in a connected component of cannot-links' complementary graph can grow exponentially $O\left(3^{|r|/3}\right)$, where $|r|$ is the size of cannot-links' complementary graph. In our experiments (see Section 5), when 1/20 constraints in Table 2 are used, $Q^{(r)}$ are 992 and 432 on camera and phone data sets, respectively; when 1/5 constraints are used, $Q^{(r)}$ grow to 3,019,414 and 3,254,972, and then the program, downloaded from [5] authors' website[1], crashed our server computer (2 Quad-Core AMD Opteron Processors, 2.70 GHz, 16GB Memory).

Synonyms Grouping: In [12], the authors proposed a method based on several similarity metrics to map discovered feature expressions to features in a given feature taxonomy of a domain. This is very different from our work as we do not have predefined feature taxonomy. The proposed method produces groupings automatically. [13] grouped product features using WordNet synonyms with

[1] http://pages.cs.wisc.edu/ãndrzeje/research/df_lda.html

poor results. [14] extracted and clustered semantic properties of reviews based on pros/cons annotations, which is different from our work of grouping product features (also we do not have pros/cons). In [15], a semi-supervised learning method is used. However, it requires the user to provide labeled examples, whereas this study does not need any pre-labeled examples. It thus solves a different problem. In [6], product features were grouped using a multilevel latent semantic association technique, called mLSA. At any level of their multilevel algorithm the original LDA is directly applied. We propose *constrained-LDA*.

3 The Proposed Algorithm

The original LDA is a purely unsupervised model, ignoring any pre-existing domain knowledge. However, as it is known in the semi-supervised clustering research [2, 3], the pre-existing knowledge can guide clustering algorithms to produce better and/or more meaningful clusters. We believe that they can help LDA as well, which is essentially a clustering algorithm. In our application domain, the prior knowledge about product features can help group domain synonym features, as explained in Section 1. In this section, we first give an introduction to LDA and then present the proposed *constrained-LDA* which can use pre-existing knowledge expressed as must-link and cannot-link constraints.

3.1 Introduction to LDA

A variety of probabilistic topic models have been proposed, and LDA is one of the most popular topic modeling methods [16]. Similar to other methods, LDA's input is a term × document matrix, and it outputs the document-topic distribution θ and topic-word distribution ϕ. In order to obtain the distributions θ and ϕ, two main algorithms were proposed, EM [7] and Gibbs Sampling [8]. In this study, we use the Gibbs Sampling. For Gibbs sampling based LDA, the most important process is the updating of topic for each term in each document according to the probabilities calculated using Equation 1.

$$\mathrm{P}\left(z_i = k | w_i = v, \mathbf{z}_{-i}, \mathbf{w}_{-i}\right) = \frac{\mathbf{C}_{vk}^{WT} + \beta}{\sum_v \mathbf{C}_{vk}^{WT} + V\beta} \frac{\mathbf{C}_{dk}^{DT} + \alpha}{\sum_k \mathbf{C}_{dk}^{DT} + K\alpha} \tag{1}$$

where $z_i = k$ represents the assignment of the i^{th} term in a document to topic k, $w_i = v$ represents that the observed term w_i is the v^{th} term in the vocabulary of the text corpus, and $\mathbf{z}_{-i}$ represents all the topic assignments excluding the i^{th} term. $\mathbf{C}_{vk}^{WT}$ is the number of times that term v is assigned to topic k, and $\mathbf{C}_{dk}^{DT}$ is the number of times that topic k has occurred in document d. Furthermore, K is the number of topics (which is an input given by the user), V is the size of the vocabulary, α and β are the hyper-parameters for the document-topic and topic-word Dirichlet distributions, respectively. (α and β are set to 50/K and 0.01 by default.) After N iterations of Gibbs sampling for all terms in all documents,

document-topic distribution θ and topic-word distribution ϕ are finally estimated using Equations 2 and 3.

$$\theta_{dk} = \frac{\mathbf{C}_{dk}^{DT} + \alpha}{\sum_k \mathbf{C}_{dk}^{DT} + K\alpha} \tag{2}$$

$$\phi_{vk} = \frac{\mathbf{C}_{vk}^{WT} + \beta}{\sum_v \mathbf{C}_{vk}^{WT} + V\beta} \tag{3}$$

3.2 Constrained-LDA

For constrained-LDA, constraints from the existing knowledge are added, and each term in the constraints is assumed to belong to only one topic. Compared with LDA, *constrained-LDA* has 2 more inputs, a set of must-link constraints and a set of cannot-link constraints. The main idea of the proposed approach is to revise the topic updating probabilities computed by LDA using the probabilities induced from the constraints. That is, in the topic updating process (shown in Equation 1), we compute an additional probability $q(z_i = k)$ from the must-links and cannot-links for every candidate topic in 1, 2,, K, and then multiply it to the probability calculated by the original LDA model as the final probability for topic updating (see Equation 4).

$$\mathrm{P}\left(z_i = k | w_i = v, \mathbf{z}_{-i}, \mathbf{w}_{-i}\right) = q(z_i = k) \frac{\mathbf{C}_{vk}^{WT} + \beta}{\sum_v \mathbf{C}_{vk}^{WT} + V\beta} \frac{\mathbf{C}_{dk}^{DT} + \alpha}{\sum_k \mathbf{C}_{dk}^{DT} + K\alpha} \tag{4}$$

As illustrated by Equations 1 and 4, $q(z_i = k)$ plays a key role in *constrained-LDA*, because $q(z_i = k)$ represents intervention or help from pre-existing knowledge of must-links and cannot-links. In this study, $q(z_i = k)$ is computed as follows: For the given term w_i, if w_i is not constrained by any must-links or cannot-links, $\{q\left(z_i = k\right) | k = 1, \ldots, K\}$=1; otherwise, $\{q\left(z_i = k\right) | k = 1, \ldots, K\}$ is calculated using the following 4 steps in Figures 1 and 2.

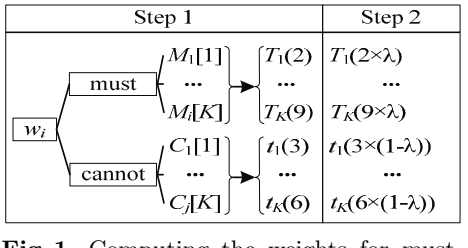


Fig. 1. Computing the weights for must-topics and cannot-topics

Input: w_i;
w_i's must-topics' weights: weight(w_i, T_k), k=1,2,…,K;
w_i's cannot-topics' weights: weight(w_i, t_k), k=1,2,…,K;
Output: $\{q(z_i=k)|k=1,2,\ldots,K\}$

1. Initial all $\{q(z_i=k)|k=1,2,\ldots,K\}$ to zero
2. **//Step 3 -** Aggregate
3. **for** (k **in** {1,2,…,K})
4. **if** (k **in** { w_i's must-topics }) $q(z_i=k)$ += weight(w_i, T_k)
5. **if** (k **in** { w_i's cannot-topics }) $q(z_i=k)$ -= weight(w_i, t_k)
6.
7. **//Step 4 -** Normalize and relax
8. max = $\{q(z_i=k)|k=1,2,\ldots,K\}_{\max}$
9. min = $\{q(z_i=k)|k=1,2,\ldots,K\}_{\min}$
10. **for** (k **in** {1,2,…,K})
11. $q(z_i = k) = \frac{q(z_i=k)-\min}{\max-\min}$
12. $q(z_i = k) = q(z_i = k) \times \eta + (1 - \eta)$

Fig. 2. Probability aggregation and relaxation

Step 1 - get the must-topics weight and cannot-topics weight of w_i. Here must-topics mean the topics that the term w_i should be grouped into, while cannot-topics mean the topics that the term w_i should not be grouped into. For the given term w_i, its must-linked and cannot-linked terms are first found by querying must-links and cannot-links stores. Second, the topics of these terms are further obtained from the topic modeling. Then, we can obtain w_i's must-topics and cannot-topics weights.

For example, w_i's must-linked and cannot-linked terms are M_1, M_2 and C_1, C_2, C_3 respectively. Furthermore, M_1, M_2 and C_1, C_2, C_3 are assigned to topic k by LDA (denoted by $M_1[k]$, $M_2[k]$ and $C_1[k]$, $C_2[k]$, $C_3[k]$). So, for topic k, w_i's must-topics and cannot-topics weights are weight(w_i, T_k ($|\{M_1, M_2\}|$))=weight (w_i, T_k (2))=2 and weight(w_i, t_k ($|\{C_1, C_2, C_3\}|$))=weight(w_i, t_k (3))=3, respectively. Here, weight(w_i, T_k) or weight(w_i, t_k) is the weight that w_i should or should not be assigned to topic k; T_k (2) represents there are 2 linked terms being assigned to topic k in the must category, and t_k (3) represents there are 3 linked terms being assigned to topic k in the cannot category.

Step 2 - adjust the relative influences between must-link category and cannot-link category. In extracting the two types of constraints, the qualities of must-links and cannot-links may be different from each other. We use a damping factor λ to adjust the relative influences based on the constraint qualities. Specifically, all the must-topics' weights are multiplied by λ, while the cannot-topics' weights are multiplied by (1- λ).

Following the above example, T_k (2) is adjusted to T_k ($2 \times \lambda$) while t_k (3) to t_k ($3 \times (1 - \lambda)$). In this study, the default value of λ is empirically set to 0.3.

Based on the results of above two steps, Steps 3 and 4 are further proposed to convert the weights of must-topics and cannot-topics to $\{q(z_i = k) | k = 1, \ldots, K\}$, as shown in Figure 2.

Step 3 - aggregate the weights for each candidate topic. For the given term w_i, its candidate topics can fall into one of the three types, must-topics, unconstrained topics and cannot-topics. Recall must-topics mean the topics that w_i should be assigned to while cannot-link means the topics that w_i should not be assigned to. Thus, for calculating the probability that w_i will be assigned to candidate topic k, if k is in must-topics, we add $weight(w_i, T_k)$ to $q(z_i = k)$ in order to enhance the probability that w_i is assigned to topic k; if k is in cannot-topics, we subtract $weight(w_i, t_k)$ to $q(z_i = k)$ in order to decrease the probability that w_i is assigned to topic k (lines 2 to 6 in Figure 2).

In the above example, for the candidate topic k, the weight $q(z_i = k)$ is: $0 + weight(w_i, T_k(2 \times \lambda)) - weight(w_i, t_k(3 \times (1 - \lambda))) = 2 \times \lambda - 3 \times (1 - \lambda)$.

Step 4 - normalize and relax the weight of each candidate topic. Since the constraints are not guaranteed to be correct especially when the constraints are extracted automatically, there should be a parameter to adjust the constraint's strength to the model according to the quality of the constraints. When the constraints are completely correct, the model should treat these constraints as hard-constraints; when the constraints are all wrong, the model should discard

them. In order to achieve this aim, $\{q(z_i = k) | k = 1, \ldots, K\}$ are adjusted by the relaxation factor as follows:

Before being relaxed, $\{q(z_i = k) | k = 1, \ldots, K\}$ are normalized to [0, 1] using Equation 5 (lines 8 to 11 in Figure 2). In Equation 5, max and min represent the maximum and minimum values of $\{q(z_i = k) | k = 1, \ldots, K\}$, respectively.

$$q(z_i = k) = \frac{q(z_i = k) - \min}{\max - \min} \tag{5}$$

Then, $\{q(z_i = k) | k = 1, \ldots, K\}$ are relaxed by the relaxation factor based on Equation 6 (line 12 in Figure 2). The default value of is set to 0.9 in our study (see the evaluations in Section 5.6).

$$q(z_i = k) = q(z_i = k) \times \eta + (1 - \eta) \tag{6}$$

Note that, for our application of grouping product features, each product feature is considered as a term. Moreover, only ϕ needs to be estimated by Equation 3 to output a set of topics and each topic contains a set of terms which belong to the topic.

4 Constraint Extraction

We now come back to our application and discuss how to extract constraints automatically. The general idea has been discussed earlier. For completeness, we briefly discuss them here again.

Must-link: If two product features f_i and f_j share one or more words, we assume them to form a must-link, i.e., they should be in the same topic, e.g., "battery power" and "battery life".

Cannot-link: If two product features f_i and f_j occur in the same sentence and they are not connected by "and", the two features form a cannot-link. The reason for this is that people usually do not repeat the same feature in the same sentence. Features linked by "and" are not used as our experience showed that "and" can be quite unsafe. It frequently links features from the same topic, especially product names based features.

5 Experimental Evaluation

In this Section, we evaluate the proposed constrained-LDA model in a variety of settings, and compare it with the original LDA and the recent multilevel mLSA. We do not compare with the similarity based method in [12] because their technique requires a given feature taxonomy, which we do not use.

5.1 Data Sets

In order to demonstrate the generality of the proposed algorithm, experiments have been conducted in two domains: digital camera and cell phone. We used

two data sets with feature annotations from the Customer Review Datasets[2], which have been widely used by researchers for opinion mining. We selected the reviews for digital cameras and cell phones. Their feature annotations are used in our system. Since these two data sets are too small for topic modeling, we crawled many other camera and phone reviews from Amazon.com. The details of each data set are given in Table 1.

Table 1. Summary of the data sets

	#Reviews	#Sentences	#Vocabulary
Camera	2,400	20,628	7,620
Phone	1,315	18,393	7,376

5.2 Gold Standard

Since the product features in the Customer Review Datasets have already been annotated by human annotators, these annotated product features are grouped manually to form a gold standard for each data set. For the digital camera data set, we group the features into 14 topics, according to the camera's taxonomy published by Active Sales AssistantTM, a product of Active Decisions, which is available at www.activebuyersguide.com [12]. For the cell phone data set, the topics published by Google products are adopted, and all the cell phone features are grouped into 9 topics.

5.3 Evaluation Measure

The performance of our product features grouping algorithm is evaluated using Rand Index [17], which has been used by several researchers [14, 18, 3]. Rand Index is also the evaluation measure used in [6].

5.4 Compared with LDA

[5] proposed the most recent LDA model (called *DF-LDA*) that can consider must-link and cannot-link constraints. However, as explained in Section 2.1, *DF-LDA* cannot process a large number of constraints. Due to *DF-LDA*'s limitation, we only report the comparison results with the original LDA. Both the original LDA and the proposed *constrained-LDA* were run using different numbers of topics, 20, 40, 60, 80, 100 and 120, in the two domains. Note that LDA requires the number of topics to be specified by the user. Note also we do not report the results of using the original numbers of topics (14 and 9) for the two data sets as they were poorer (see the trends in Figure 3). Using only must-links, only cannot-links, and their combination were all experimented. The number of constraints extracted from each data set is given in Table 2. All the results are shown in Figure 3.

[2] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Table 2. Number of the extracted constraints

	#Must-links	#Cannot-links
Camera	300	5172
Phone	184	5009

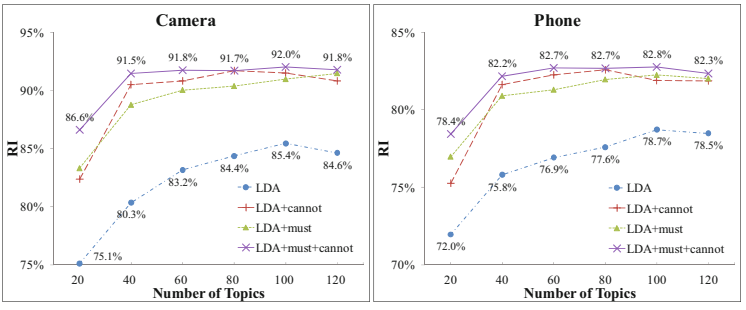


Fig. 3. RI results of constrained-LDA and the original LDA

From Figure 3, we can see that the patterns are about the same for different methods on different data sets, which show that the results are consistent. Below we make some additional observations: (*1*)All the constrained methods (*LDA+cannot*, *LDA+must* and *LDA+must+cannot*) perform much better than the original LDA model (LDA). For smaller numbers of topics, the improvements were more than 10% for the digital camera corpus, and around 7% for the cell phone corpus. With more topics, the improvements are slightly less, but still 7% for the digital camera and 4% for the cell phone. (*2*)Both cannot-links (*LDA+cannot*) and must-links (*LDA+must*) perform well, although cannot-links are slightly more effective than must-links on average. This phenomenon indicates that our assumption about cannot-link is reasonable and the quality of the extracted cannot-links is good. When the number of topics is small or large, the must-links are slightly better than cannot-links. We believe the reason is that in these two ends, cannot-link terms were either forced into the same topics (for a small number of topics), or easily spread into too many topics. The original LDA also shows this behavior, which is fairly easy to understand. (*3*)The combination of must-links and cannot-links (*LDA+must+cannot*) consistently outperforms each individual type of constraints alone (*LDA+cannot* and *LDA+must*). Although the margins of improvements were not very large, they were consistent. This also indicates that the must-link and cannot-link constraints are already quite effective individually.(*4*)In practice, it is often more effective to use a smaller number of topics, which are easy to understand and to handle by the users. In both cases, 40 topics seem to be optimal.

In summary, we can see that unsupervised topic modeling can be improved by adding must-link and cannot-link constraints. Note that each feature expression is considered as a term in all our experiments.

5.5 Comparing with mLSA

As mentioned earlier, the recent multilevel latent semantic association method *mLSA* [6] solves the same problem as we do. [6] shew that *mLSA* performs better than the existing methods, e.g., LDA-based and Kmeans-based algorithm. We thus only compare the proposed *constrained-LDA* with *mLSA*, but not other existing methods. The comparisons are made based on both the digital camera corpus and the cell phone corpus. The results are shown in Figure 4. We only used 40 topics, which appeared to be the optimal number among our tested topic numbers in Figure 3.

As demonstrated in Figure 4, *mLSA*(2: red bar) achieves encouraging results by transforming the input document content before applying LDA. Our *constrained-LDA* model does not make any efforts to re-organize or transform the input document content, and our input is the set of original reviews. However, the results produced by *constrained-LDA*(*LDA+cannot*, *LDA+must* and *LDA+must+cannot*) are all substantially better than those of *mLSA*. This observation shows the positive influence of constraints.

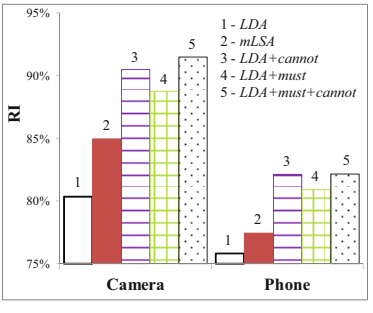


Fig. 4. Comparisons with the existing methods

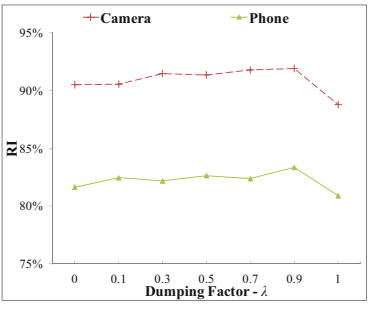


Fig. 5. λ's influence on the overall performance

5.6 Influence of Parameters

Compared to the original LDA model, the proposed *constrained-LDA* has two additional parameters, i.e., damping factor λ and relaxation factor η, as mentioned in Section 3.2. In this section, we discuss their influences to the overall performance.

Influence of the damping factor - λ: Recall that damping factor λ is used to adjust the relative influences of must-links and cannot-links on the proposed model. In Figure 5, λ=0 means the proposed model is only constrained by cannot-links, whereas λ=1 means that the proposed model is only constrained by must-links. That is, larger λ values mean more influences of must-links and less influence of cannot-links. As shown in Figure 5, with increased influence of must-links over cannot-links, the performance of constrained-LDA improves slightly. However, when there is only must-links (λ=1), the performance drops sharply to the lowest point. This illustrates the synergetic effect of must-links and cannot-links:

they help each other. Since the λ values after 0.3 produce very similar results, we used λ=0.3 as the default for λ. The experimental results in Figures 3 and 4 all used this default damping factor.

Influence of the relaxation factor - η: In this study, the relaxation factor η represents the strength of the constraints on the LDA model. When η=0, it means that no constraint is added to the LDA model. Then, *constrained-LDA* reduces to the original LDA. When η=1, it means that both must-link and cannot-link constraints become hard constraints and cannot be violated. The influence of η on the overall performance is shown in Figure 6.

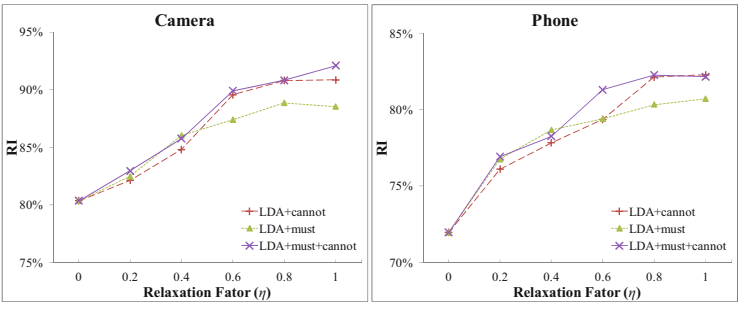


Fig. 6. η's influence on the overall performance (#topics = 40)

As shown in Figure 6, with the growth of the strength of the constraints, the performances of *LDA+cannot*, *LDA+must* and *LDA+must+cannot* increase considerably. This observation not only shows that the constraints clearly help the performances of topic modeling (or LDA), but also shows that the qualities of the extracted must-links and cannot-links are quite good, especially the extracted cannot-links. In fact, using both must-link and cannot-link constraints, when η= 1, the results are the best for both the digital camera data and cell phone data. We use η=0.9 as the default in the system as in general one may not be able to extract very high quality constraints. In our experiments reported earlier in Figures 3, 4 and 5, the default η= 0.9 was used.

6 Conclusions

This paper enhanced the popular topic modeling method LDA with the ability to consider existing knowledge in the form of must-link and cannot-link constraints. In our application, we experimented with two opinion mining data sets to group product feature synonyms, and the proposed *Constrained-LDA* outperformed the existing methods by a large margin, which showed that constraints as prior knowledge can help unsupervised topic modeling. Moreover, this paper also proposed two methods to extract the two types of constraints automatically. Experimental results showed that their qualities were high (see Figure 6).

References

1. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of SIGKDD, pp. 168–177 (2004)
2. Basu, S., Davidson, I., Wagstaff, K.: Constrained clustering: Advances in algorithms, theory, and applications. Chapman & Hall/CRC, Boca Raton (2008)
3. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: Proceedings of ICML, pp. 577–584 (2001)
4. Andrzejewski, D., Zhu, X.: Latent Dirichlet Allocation with topic-in-set knowledge. In: Proceedings of NAACL HLT, pp. 43–48 (2009)
5. Andrzejewski, D., Zhu, X., Craven, M.: Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In: Proceedings of ICML, pp. 25–32 (2009)
6. Guo, H., Zhu, H., Guo, Z., Zhang, X., Su, Z.: Product feature categorization with multilevel latent semantic association. In: Proceedings of CIKM, pp. 1087–1096 (2009)
7. Blei, D., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3(3), 993–1022 (2003)
8. Griffiths, T., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences 101(Suppl 1), 5228–5535 (2004)
9. Blei, D., McAuliffe, J.: Supervised topic models. Advances in Neural Information Processing Systems 20, 121–128 (2008)
10. Ramage, D., Hall, D., Nallapati, R., Manning, C.: Labeled, LDA: A supervised topic model for credit attribution in multi-labeled corpora. In: Proceedings of EMNLP, pp. 248–256 (2009)
11. Chang, J., Blei, D.: Relational topic models for document networks. In: Proceedings of the 12th International Conference on Artificial Intelligence and Statistics(AISTATS), Clearwater Beach, Florida, USA (2009)
12. Carenini, G., Ng, R., Zwart, E.: Extracting knowledge from evaluative text. In: Proceedings of International Conference on Knowledge Capture, pp. 11–18 (2005)
13. Liu, B., Hu, M., Cheng, J.: Opinion Observer: Analyzing and Comparing Opinions on the Web. In: Proceedings of WWW, pp. 342–351 (2005)
14. Branavan, S.R.K., Chen, H., Eisenstein, J., Barzilay, R.: Learning document-level semantic properties from free-text annotations. In: Proceedings of ACL, pp. 569–603 (2008)
15. Zhai, Z., Liu, B., Xu, H., Jia, P.: Grouping Product Features Using Semi-supervised Learning with Soft-Constraints. In: Proceedings of COLING (2010)
16. Steyvers, M., Griffiths, T.: Probabilistic topic models. In: Handbook of Latent Semantic Analysis, pp. 424–440 (2007)
17. Rand, W.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66(336), 846–850 (1971)
18. Cardie, C., Wagstaff, K.: Noun phrase coreference as clustering. In: Proceedings of the Eleventh National Conference on Artificial Intelligence, pp. 82–89 (1999)

Semantic Dependent Word Pairs Generative Model for Fine-Grained Product Feature Mining

Tian-Jie Zhan and Chun-Hung Li

Department of Computer Science
Hong Kong Baptist University
{tjzhan,chli}@comp.hkbu.edu.hk

Abstract. In the field of opinion mining, extraction of fine-grained product feature is a challenging problem. Noun is the most important features to represent product features. Generative model such as the latent Dirichlet allocation (LDA) has been used for detecting keyword clusters in document corpus. As adjectives often dominate review corpus, they are often excluded from the vocabulary in such generative model for opinion sentiment analysis. On the other hand, adjectives provide useful context for noun features as they are often semantically related to the nouns. To take advantage of such semantic relations, dependency tree is constructed to extract pairs of noun and adjective with semantic dependency relation. We propose a semantic dependent word pairs generative model for pairs of noun and adjective for each sentence. Product features and their corresponding adjectives are simultaneously clustered into distinct groups which enable improved accuracy of product features as well as providing clustered adjectives. Experimental results demonstrated the advantage of our models with lower perplexity, average cluster entropies, compared to baseline models based on LDA. Highly semantic cohesive, descriptive and discriminative fine-grained product features are obtained automatically.

Keywords: Product feature mining, semantic dependency, generative model.

1 Introduction

Challenge in opinion mining is to identify the precise object on which the opinion is expressed that is finding fine-grained product features from the user reviews. For example, "I do not like the camera with low memory capacity.", the author expressed its negative opinion on the camera memory capacity rather than the camera as a whole, but the classical approach will classify the sentence into negative category. To overcome the shortcomings, feature-based opinion mining [1] proposed to extract such a pair of noun and adjective that is adjacent to each other so that noun can indicate a product feature and adjective indicates the opinion orientation. But standard approaches take each noun as a unique product feature rather than categorize the nouns into clusters by their semantic

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 460–475, 2011.

relevance to each other so that each of the clusters could represent a discriminative product feature. To extract fine-grained product feature, similarity measures between the nouns are designed based on manual annotated tags and ontology dictionary such as WordNet [3,4]. Then classical machine learning approaches are used to categorize the nouns into different product features. In text processing domain, generative models are efficient for document clustering and word clustering, e.g., latent Dirichlet allocation (LDA) [10]. A local version of LDA [3] has been proposed to categorize the noun words as the product feature. A multi-level latent semantic association model [4] extracted fine-grained product features by exploring the sliding window based context of each noun but it does not consider semantic relevance between the noun and the words in the context. Though there is some work [6] taking advantage of semantic relations for sentiment classification, most of the previous works [1,2,3] on extracting product features only take account of nouns or with a little bit utilization of non-nominal terms such as adjectives. Even if non-nominal terms are considered, but their semantic relevance to the nouns gets less attention. To utilize the non-nominal terms, a typical approach [5] defined the noun's context features as its co-occurring words in noun phrases and then use a graph clustering method on the co-occurrence relations between nouns and their context features to categorize the noun words. Another similar approach [7] is to apply non-negative matrix factorization method on the co-occurrence of nouns and their semantic relevant context features. But both of them disregard the direct co-occurrence between nouns. To combine the co-occurrence of nouns and semantic dependencies between nouns and other terms, this paper contributes to propose a semantic dependent word pair generative model to take advantage of nouns, adjectives and their semantic relevance for the extraction of fine-grained product features.

2 Data Representation and Problem Definition

Bag-of-word is a popular representation of document in text processing but is not sufficient for feature-based opinion mining [2]. To extract relation between product feature and opinions on the feature, noun-adjective pairs are extracted from sentences in [1,2]. In their work, moving window method are used to capture text pieces following a pattern where the word at window' center is a noun and adjectives within the window are opinionated terms to evaluate the product feature indicated by the noun. For example, it can be extracted that pairs such as ⟨ "restaurant", "good"⟩, ⟨"service", "friendly"⟩ and ⟨"price", "reasonable"⟩ from the sentence shown in Fig.1. But syntactical approaches may extract noisy pairs which do not have a dependency relation. For example, bad pairs like ⟨"food quality" ,"friendly"⟩ and ⟨"service", "reasonable"⟩ would be included by moving window of size two, which may deliver negative effect on extraction of product feature. Another problem is that unigram noun is not sufficient to represent a product feature such as "food quality" as shown in Fig.1.

A dependency tree is represented as a set of dependencies $\{w_i^h, w_i^m, r_i\}$ where head node w_i^h and modifier node w_i^m forms a head-modifier dependency of r_i

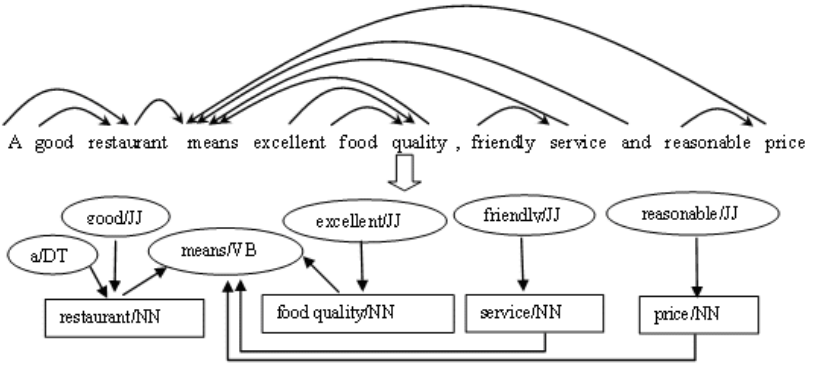


Fig. 1. Illustration of dependency tree and semantic neighborhood structure. Top is the dependency tree of a sentence and bottom is the associated semantic neighbors structure where square box contains the noun fragments and elliptical box is the semantic neighbors of associated noun fragments with arrow represents dependency.

type relation as shown in the top half of Fig.1. So the noun compound is defined with syntactical rules by largest sequence of consecutive noun words with each word tagged as *NN* by parts-of-speech (POS) or such a noun sequence including word "of". By merging each word in a noun compound, the dependency tree obtains a new node representing the noun compound, denoted as noun node. And semantic neighbors of noun node are the nodes with adjacent dependency relation to the associated noun compound in the dependency tree, e.g., "good" and "means" as the semantic neighbor of "restaurant" in Fig.1. Thereafter a semantic structure comprised of noun compound and their semantic neighbor nodes can form a semantic neighborhood structure, e.g., the bottom half in Fig.1 where an arrow suggests a neighboring relation between a noun fragment and a semantic neighbor with associated POS tag following the lexicons for each node. Here POS Tag is specific to denote the noun fragments instead of noun word only.

We use word "noun" to denote noun or noun compounds discussed above and "neighbor" is used to denote the semantic neighbor. To concentrate on the generative process to produce pairs of noun and neighbor, only the most typical neighbors are considered, including neighbors with POS tag "JJ", "JJR","JJS", "VBN" and "VBG", namely, adjectives and adjectival-verbs, as shown in

Table 1. Noun-neighbor pairs generation rules from dependency tree of a sentence

Sub-tree of the semantic neighbors structure	**Pairs generated**
$\langle w_H/\text{NN}, w_M/Tag\rangle$: Tag=JJ|JJS|JJR|VBN|VBG	$\langle w_H\ , w_M\rangle$
$\langle w_R/\text{VB}, w_H/\text{NN}\rangle$, $\langle w_R/\text{VB}, w_J/Tag\rangle$: Tag=JJ|JJS|JJR, w_R is a link verb	$\langle w_H\ , w_J\rangle$
$\langle w_R/\text{MD}, w_H/\text{NN}\rangle$, $\langle w_R/\text{MD}, w_V/\text{V}\rangle$, $\langle w_V/\text{VB}, w_J/Tag\rangle$: Tag=JJ|JJS|JJR, w_R is a link verb	$\langle w_H\ , w_J\rangle$

Table 1. The resultant semantic dependent word pairs from Fig.1 are ⟨"restaurant", "good"⟩, ⟨"food quality", "excellent"⟩, ⟨"service", "friendly"⟩ and ⟨"price", "reasonable⟩. Given a collection of sentences related to some product domain such as "restaurant", "hotel", the sentences are assumed to be preprocessed into a triple stream $\{\langle d_i, w_i^n, w_i^s\rangle\}$ where d_i, w_i^n, w_i^s are the sentence index, noun phrase drawn from a vocabulary V_{NN} and the associated semantic neighbors drawn from vocabulary V_{SNei} respectively. Also a sentence s can be represented as a set of pairs $\{ w_i^n, w_i^s | d_i{=}s\}$. Given the processed data, now the problem is given as below.

Problem Definition: Given the triple stream $\{\langle d_i, w_i^n, w_i^s\rangle\}$, the problem is how to extract representative and semantic relevant lexicons from the noun vocabulary V_{NN} and adjective vocabulary V_S to represent and distinguish different fine-grained product features.

Different with the unigram representation, our problem should deal with pairs of noun and adjective with semantic relevance, from which cluster of nouns and adjectives should be obtained for each fine-grained product feature. Unlike previous work [3] to formalize the noun words clustering as a bi-clustering problem and appeal to LDA, our problem is to generate words from two independent vocabularies of noun and adjective. Our institution is to extend the LDA into a "variant" with semantic dependency between pairs of Part-of-speech tokens, which will be discussed in detail in the following sections.

3 Semantic Dependent Word Pair Generative Model

To extract the fine-grained product feature, we propose our approach with a generative mode. The semantic dependent word pair generative model (SDWP) can be regarded as an extension of Latent Dirichlet Allocation (LDA). Each cluster learned by the model would be regarded as a product feature. The original LDA models document as a mixture of topics (clusters) with each topics comprised of words in a vocabulary. As sentence based analysis is more suitable for fine-grained product feature extraction, LDA mentioned in this paper will be referred as sentence based LDA. The basic assumption made in LDA is the exchangeability between words in a sentence. To include the semantic dependency between words into LDA, the SDWP model assumes that sentence is compose by a mixture of exchangeable nouns and another mixture of exchangeable semantic neighbors. Based on such assumption, SDWP could be proposed to generate word pairs $\{\langle w_i^n, w_i^s\rangle\}$ for each sentence.

Suppose we have a stream of triples $\{\langle d_i, w_i^n, w_i^s\rangle\}$ from N_D sentences, given hyper-parameters $H = \{\alpha, \beta^{NN}, \beta^S\}$ for Dirichlet smoothing and the pre-defined number of clusters k, generative process of SDWP can be depicted as below.

1. For each cluster f , draw *multinomial* distribution $\phi(f) \sim Dirichlet(\beta^{NN})$
2. For each cluster f , draw *multinomial* distribution $\psi(f) \sim Dirichlet(\beta^{S)}$
3. Given sentence $s = d_i$, draw *multinomial* distribution $\theta(s) \sim Dirichlet(\alpha)$

Draw a cluster index z from distribution $\theta(s)$,
Draw a noun phrase w_i^n from distribution $\phi(z)$,
Draw a semantic neighbor w_i^s from distribution $\psi(z)$,
Get triple $\langle s, w_i^n, w_i^s w\rangle$.

The graphical model of SDWP can be illustrated in Fig.2. Recalling the graphical model definition in Fig.2, given the cluster index z, the sentence index s are independent with w_i^n and w_i^s respectively as in the case of LDA. It is consist with the results in [7] that w_i^n and w_i^s are independent conditional on the cluster index.

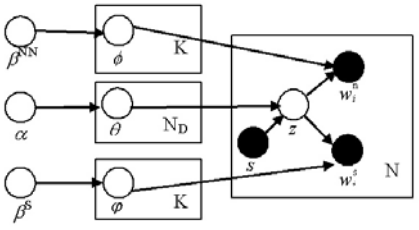


Fig. 2. Graphical model of SDWP where circle nodes represent the variables of the model with the filled black one as the observed variables. N is the total number of such triples.

4 Inference and Parameter Estimation

As described above about the SDWP, given the hyper-parameter H and a collection of sentences $S = \{d_s\}_s$, we can calculate the evidence likelihood of tokens streams comprised by noun-neighbor pairs as $W=\{\langle w_i^n, w_i^s\rangle\}|_{i=1:1:N}$ below. The sentence index for each pair $\langle w_i^n, w_i^s\rangle$ is denoted by d_i. We would start the study of the probability of the model from the parameters $\theta=\{\theta(s)\}_s$, $\phi=\{\phi(i)\}_i$ and $\psi=\{\psi(i)\}_i$ which is leading the generative process. Some word counting statistics used for inference is listed in Table 2.

Table 2. Definition of some word counts used in the inference

Denotation	Comment
$n(d, c)$	Counts of number of pairs assigned to cluster c in sentence d
$n(c, i, \cdot)$	Counts of assignment to cluster c of the i^{th} noun in V_{NN}
$n(c, \cdot, j)$	Counts of assignment to cluster c of the j^{th} neighbor term in V_S

As shown in Fig.2, given the sentence s, z_i is drawn from $\theta(s)$. Then given the topic z_i, noun w_i^n and adjective w_i^s are drawn from $\phi(z_i)$ and $\psi(z_i)$ respectively, where for each sentence $\theta(s)$ is a K-dimension vector $\{\theta_{s,1}, \theta_{s,2}, \ldots, \theta_{s,K}\}$ drawn from a Dirichlet distribution conditional on hyper-parameter $\alpha=\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ as in (1):

$$p(\theta(s) \mid H) = p(\theta(s) \mid \alpha) = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \theta_{s,1}^{\alpha_1}, \theta_{s,2}^{\alpha_2}, \ldots, \theta_{s,K}^{\alpha_K}. \tag{1}$$

where Γ is Gamma function, α_i is positive meanwhile $\theta_{s,i} > 0$ and subjected to $\sum_i \theta_{s,i}$=1 .

Similarly, given parameter H with $\beta^{NN} = \{\beta_1^{NN}, \beta_2^{NN}, \ldots, \beta_{L_{NN}}^{NN}\}$ and $\beta^S = \{\beta_1^S, \beta_2^S, \ldots, \beta_{L_S}^S\}$, the probability density function of ϕ and ψ can be computed as in (2) and (3),

$$p(\phi \mid H) = p(\phi \mid \beta^{NN}) = \prod_{i=1}^{K} \frac{\Gamma(\sum_{j=1}^{L_{NN}} \beta_j^{NN})}{\prod_{j=1}^{L_{NN}} \Gamma(\beta_j^{NN})} \phi_{i,1}^{\beta_1^{NN}}, \phi_{i,2}^{\beta_2^{NN}}, \ldots, \phi_{i,L_{NN}}^{\beta_{L_{NN}}^{NN}}, \tag{2}$$

$$p(\psi \mid H) = p(\psi \mid \beta^S) = \prod_{i=1}^{K} \frac{\Gamma(\sum_{j=1}^{L_S} \beta_j^S)}{\prod_{j=1}^{L_S} \Gamma(\beta_j^S)} \psi_{i,1}^{\beta_1^S}, \psi_{i,2}^{\beta_2^S}, \ldots, \psi_{i,L_S}^{\beta_{L_S}^S}. \tag{3}$$

where $\phi_i = \{\phi_{i,1}, \phi_{i,}, \ldots, \phi_{i,L_{NN}}\}$ and $\psi_i = \{\psi_{i,1}, \psi_{i,}, \ldots, \psi_{i,L_S}\}$ are the parameters of *multinomial* distribution of the i^{th} cluster, associated with noun and semantic neighbor vocabulary respectively, with constraints of $\sum_j \phi_{i,j} = 1$ and $\sum_j \psi_{i,j} = 1$ while each $\phi_{i,j}$ and $\psi_{i,j}$ are positive, for each cluster i. Here L_{NN} and L_S are the size of nouns vocabulary V_{NN} and semantic neighbor terms vocabulary V_S respectively.

Given the parameters θ, for each associated pair $\langle w_i^n, w_i^s \rangle$, it can be drawn the latent variables z_i indicating which cluster the pair belong to. The stream of $\{z_i\}$ is denoted by Z. Based on the model illustrated in Fig.2, the joint probability of Z and W and the parameters $\{\theta, \phi, \psi\}$ can be written as

$$p(W, Z, \theta, \phi, \psi \mid H) = p(\theta \mid H) p(\phi \mid H) p(\psi \mid H) \prod_{i \in W} p(z_i \mid \theta, s) p(w_i^n \mid z_i, \phi) p(w_i^s \mid z_i, \psi). \tag{4}$$

From the *multinomial* distribution with parameter $\theta(s)$, it is easy to know that $p(z_i \mid \theta, s)$ equals to θ_{s,z_i}. Similarly $p(w_i^n \mid z_i, \phi)$ equals to ϕ_{z_i,w_i^n} and $p(w_i^s \mid z_i, \psi)$ equals to ψ_{z_i,w_i^s}.

Substituted with (1) (2) and (3), (4) can be rewritten as

$$p(W, Z, \theta, \phi, \psi \mid H) = C \prod_{i=1}^{K} \prod_{s \in S} \theta_{s,i}^{\alpha_i + n(s,i)} \prod_{j=1}^{L_{NN}} \phi_{i,j}^{\beta_j^{NN} + n(i,j,\cdot)} \prod_{j=1}^{L_S} \psi_{i,j}^{\beta_j^S + n(i,\cdot,j)}. \tag{5}$$

where C is the normalized factor calculated as

$$C = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \frac{\Gamma(\sum_{j=1}^{L_{NN}} \beta_j^{NN})}{\prod_{j=1}^{L_{NN}} \Gamma(\beta_j^{NN})} \frac{\Gamma(\sum_{j=1}^{L_S} \beta_j^S)}{\prod_{j=1}^{L_S} \Gamma(\beta_j^S)}. \tag{6}$$

With the probability discussed above, we can infer the latent variable $\{z_i\}$ and parameters H with details in the subsections.

4.1 Latent Variable Inference

Similar to LDA, the latent variables is intractable for inference. So approximating approaches have been proposed to infer the latent variables with LDA such as Gibbs sampling and variant method. In this paper, we employ collapse Gibbs sampling for inference of the latent variables. Parameters $\{\theta, \phi, \psi\}$ could be integrated out on the joint probability (6) as

$$p(W,Z|\ H) = \int\int\int C \prod_{i=1}^{K} \prod_{s \in S} \theta_{s,i}^{\alpha_i + n(s,i)} \prod_{j=1}^{L_{NN}} \phi_{i,j}^{\beta_j^{NN} + n(i,j,\cdot)} \prod_{j=1}^{L_S} \psi_{i,j}^{\beta_j^S + n(i,\cdot,j)} d\theta d\phi d\psi.$$

which can be computed as

$$p(W,Z\,|\,H) = C(\prod_{s \in S} \frac{\prod_{i=1}^{K} \Gamma(\alpha_i + n(s,i))}{\Gamma(\sum_{i=1}^{K} \alpha_i + N)})(\prod_{i=1}^{K} \frac{\prod_{j=1}^{L_{NN}} \Gamma(\beta_j^{NN} + n(i,j,\cdot))}{\Gamma(\sum_{j=1}^{L_{NN}} \beta_j^{NN} + N)})(\prod_{i=1}^{K} \frac{\prod_{j=1}^{L_S} \Gamma(\beta_j^S + n(i,\cdot,j))}{\Gamma(\sum_{j=1}^{L_S} \beta_j^S + N)}). \tag{7}$$

In Gibbs sampler, the essence is to randomly reserve one variable for sampling while assuming the others are the true samples drawn from the model. We denote z^{-i} as subset of Z excluding z_i. The posterior probability of z_i conditioned on H,W and z^{-i} can be compute as

$$p(z_i = k\,|\,z^{-i}, W, H) = \frac{n(d_i,k) - 1 + \alpha_k}{\sum_{j=1}^{K} \alpha_j + N - 1} \frac{n(k, w_i^n, \cdot) - 1 + \beta_{w_i^n}^{NN}}{\sum_{j=1}^{L_{NN}} \beta_j^{NN} + N - 1} \frac{n(k,\cdot,w_i^s) - 1 + \beta_{w_i^s}^{S}}{\sum_{j=1}^{L_S} \beta_j^S + N - 1}. \tag{8}$$

4.2 Parameter Estimation

Similar to LDA, giving the sampling results of Z, the joint probability of parameters $\{\theta, \phi, \psi\}$ can be written as

$$p(\theta, \phi, \psi \mid\ Z, W, H) \propto C \prod_{i=1}^{K} \prod_{s \in S} \theta_{s,i}^{\alpha_i + n(s,i)} \prod_{j=1}^{L_{NN}} \phi_{i,j}^{\beta_j^{NN} + n(i,j,\cdot)} \prod_{j=1}^{L_S} \psi_{i,j}^{\beta_j^S + n(i,\cdot,j)},$$

which implies the conditional independency among θ, ϕ and ψ in between and indicates that

$$\begin{aligned}
&\theta_s \mid Z, W, H \sim \text{Dirichlet}(\alpha_1 + n(s,1), \alpha_2 + n(s,2), \ldots, \alpha_K + n(s,K)),\\
&\phi_i \mid Z, W, H \sim \text{Dirichlet}(\beta_1^{NN} + n(i,1,\cdot), \beta_2^{NN} + n(i,2,\cdot), \ldots, \beta_{L_{NN}}^{NN} + n(i, L_{NN}, \cdot)\\
&\psi_i \mid Z, W, H \sim \text{Dirichlet}(\beta_1^S + n(i,\cdot,1), \beta_2^S + n(i,\cdot,2), \ldots, \beta_{L_S}^S + n(i,\cdot,L_S))
\end{aligned}$$

where $\phi_i = \{\phi_{i,1}, \phi_{i,2}, \ldots, \phi_{i,L_{NN}}\}$, $\psi_i = \{\psi_{i,1}, \psi_{i,2}, \ldots, \psi_{i,L_S}\}$.

Using expectation of Dirichlet distribution, $E(Dir(\alpha)) = \langle \alpha_1, \alpha_2, \ldots, \alpha_K \rangle / \sum_i \alpha_i$, on the results obtained above, the parameters can be estimated as

$$\theta_{s,i} = \frac{\alpha_i + n(s,i)}{\sum_j \alpha_j + N}, \phi_{i,j} = \frac{\beta_j^{NN} + n(i,j,\cdot)}{\sum_t \beta_t^{NN} + n(i,t,\cdot)}, \psi_{i,j} = \frac{\beta_j^{S} + n(i,\cdot,j)}{\sum_t \beta_t^{S} + n(i,\cdot,t)}. \quad (9.1)$$

4.3 Hyper-parameter Estimation

As discussed in [14], LDA is effective and robust enough but sensitive to different setting of hyper-parameters such as α. To eliminate the influence of improper setting of hyper-parameter to the models, we follow the rules of equation (8) to update the hyper-parameter α. In this approach, hyper-parameters $\{\alpha_i\}$ is represented by α_0 and m_i where α_0 is the sums of $\{\alpha_i\}$ and m_i=α_i/α_0, where probabilistic distribution $\{m_i\}$ is assumed to be drawn from a Poisson distribution with α_0 drawn from a Gamma distribution. So given the topical variable samples from Gibbs sampling, hyper-parameters can be updated by

$$\alpha_i^{New} = \frac{\sum_{s \in S} \sum_{j=1}^{n(s,i)} 1/(\alpha_i^{old} + j - 1)}{\sum_{s \in S} \sum_{j=1}^{|s|} 1/(\alpha_i^{old} + j - 1)}, \quad (9.2)$$

where $|s|$ is the number of pairs in sentence s.

5 Evaluation

The SDWP model is approximated by Gibbs sampling method to infer the cluster index of each pairs $\langle w_i^n, w_i^s \rangle$. With the samplers drawn from Gibbs sampling process, the model's parameters can be estimated by (9.1). θ_s can be interpreted as the mixture weights of cluster occurring in sentence s. ϕ_i and ψ_i are the nouns and semantic neighbor's occurring probability distribution in the i^{th} cluster respectively. To evaluate the clustering results, three measures are used including perplexity, average clustering entropy (ACE) and normalized mutual information index (NMI) which will be discussed below.

5.1 Perplexity

Perplexity is an effective measure for generative language models. It is defined as the exponential of geometrical mean of the probability of each word-pair's occurrence. The model learned from the training corpus is denoted by $M_{Train} = \{\theta^T, \phi^T, \psi^T\}$. The likelihood of the testing sentences W_{Test} given the hyper-parameters and models can be calculated by integrating out the latent variables and the parameters as

$$p(W_{Test} \mid M_{Train}, H) = \prod_{i' \in W_{Test}} \sum_{k=1}^{K} p(z = k \mid \theta^{Test}) \phi^T_{k,w_i^n} \psi^T_{k,w_i^s}. \quad (10)$$

Here θ^{Test} is estimated by applying equation (9.1) on samples drawn from Gibbs sampling on the testing dataset. So the perplexity on joint probability of noun and semantic neighbor can be computed as

$$PPX(W_{Test}) = exp(-\frac{\sum_{i' \in W_{Test}} log(\sum_{k=1}^{K} p(z = k \mid \theta^{Test}) \phi^{T}_{k,w^{n}_{i}} \psi^{T}_{k,w^{s}_{i}})}{2 \cdot N_{Test}}). \quad (11)$$

where PPX is the perplexity function and N_{Test} is the total number of occurrence of pairs in the testing sentences W_{Test}. This perplexity is a word occurrence perplexity averaging between noun and adjective. Square of the perplexity in (11) is a perplexity of the occurrence of a word pair of noun and adjective.

To calculate the perplexity of noun features, there are two ways, including marginalize the semantic neighbor variables and calculate the likelihood of noun features conditional on semantic neighbor feature.

To represent the noun and semantic feature, W_{Test} can be rewritten as $\{W^{NN}, W^{S}\}$ where W^{NN}=$\{w^{n}_{i} : i \in W_{Test}\}$ and W^{S}=$\{w^{s}_{i}, i \in W_{Test}\}$. So the marginal perplexity and conditional perplexity can be computed as below:

$$PPX(W^{NN}) = exp(-\frac{\sum_{i' \in W_{Test}} log(\sum_{k=1}^{K} p(z = k \mid \theta^{Test}) \phi^{T}_{k,w^{n}_{i}})}{N_{Test}}). \quad (12)$$

5.2 Average Cluster Entropy

After running Gibbs sampling on the training sentences W_{Train}, the model's parameters can be estimated by (9,1) as θ^{T}=$\{\theta^{T}_{s,i}\}$, ϕ^{T}=$\{\phi^{T}_{i,j}\}$ and ψ^{T}=$\{\psi^{T}_{i,j}\}$ As discussed above, given cluster i, ϕ^{T}_{i}=$\{\phi^{T}_{i,j}\}|_{j}$ and ψ^{T}_{i}=$\{\psi^{T}_{i,j}\}|_{j}$ are the probability distribution of nouns and semantic neighbor's terms respectively. So the entropy of nouns in topic i can be computed as

$$Ent_{NN}(z = i) = -\sum\nolimits_{j} (\phi^{T}_{i,j} log(\phi^{T}_{i,j})) \quad (13)$$

Similarly the entropy of semantic neighbors for cluster i is

$$Ent_{S}(z = i) = -\sum\nolimits_{j} (\psi^{T}_{i,j} log(\psi^{T}_{i,j})) \quad (14)$$

The average cluster entropy of nouns and semantic neighbors are the mean of Ent_{NN} and Ent_{S} respectively. Low average cluster entropy means more discriminative between clusters, more cohesive within cluster and so better performance.

5.3 Normalized Mutual Information Index

The normalized mutual information (NMI) is an important external cluster evaluation measure for the density of clusters. To calculate NMI, mutual entropy between cluster and the feature should be computed in advance as $I(C, F)$ where C is the cluster variable and F is the data's features. In our case, there are two set of features for each sentences including the nouns words and semantic neighbor terms. Based on the samples of $\{z_i\}$ from the Gibbs sampling process, a cluster assignment statistics matrix A=$\{a_{i,j}\}$ can be built by taking $a_{i,j}$ as

$c(i,j)$, for noun feature, and matrix B=$\{b_{i,j}\}$ can be built with $b_{i,j}$ set as $m(i,j)$ for semantic neighbors. The mutual entropy between cluster variable and noun W^{NN} is calculated on A as

$$I(C,W^{NN}) = \sum_{i,j} \frac{c(i,j)}{|c|} log(\frac{c(i,j)|c|}{\sum_t c(i,t) \sum_t c(t,j)}), |c| = \sum_{i,j} c(i,j). \quad (15)$$

By normalization, the NMI can be computed as

$$NMI(C,W^{NN}) = I(C,W^{NN}) / \sqrt{H(C)H(W^{NN})}. \quad (16)$$

$H(C)$ and $H(W^{NN})$ are the entropy of cluster and noun features, written as $-(\sum_i s(i,\cdot) log(s(i,\cdot)/|c|))/|c|$ and $-(\sum_j s(\cdot,j) log(s(\cdot,j)/|c|))/|c|$ respectively where $s(i,\cdot)$=$\sum_j c(i,j)$ and $s(\cdot,j)$=$\sum_i c(i,j)$ could be interpreted as sum of entries in a row and column of matrix $c(\cdot,\cdot)$ respectively. Similarly, NMI between cluster variable and semantic neighbor feature can be computed by alternating matrix A with B in (14).

6 Experiments

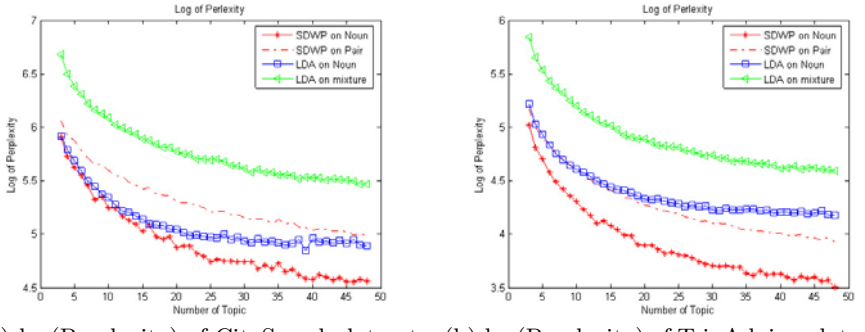
We conduct the experiments on two public datasets. "CitySearch" dataset used in [12] contains user reviews of restaurants in US, which is crawled from newyork.citysearch.com. And "TripAdvisor" dataset [13] includes user reviews on hotels, which is crawled from tripadvisor.com. Preprocessing steps include first POS tagging with OpenNLP packages [9], and then dependency parsing by Malt Parser [11]. Then a standard stop-word list [15] is used and dozens of corpus frequent terms is removed. After removal of nouns, neighbors and sentences with only one occurrence, a dictionary V_{NN} of 3377 noun fragments and a dictionary V_{SNei} of 2583 neighbors are obtained from 16456 sentences for CitySearch. The details of the two datasets are shown in Table 3.

Table 3. Summary of the dataset used in the experiment

Item / Product	#Sentence	#Word Pair	Size of V_{NN}	Size of V_{SNei}
Restaurant	16456	38751	3377	2853
Hotel	16914	44022	4024	3236

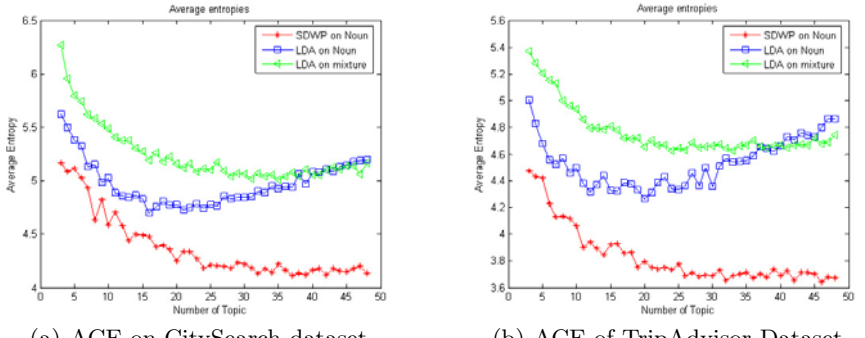
We introduce two baseline models based on conventional LDA for comparison. The first model is a sentence based unigram LDA to generate the noun occurrence regardless of the adjectives. The second baseline model takes account of the adjectives into the sentence based unigram LDA by treating nouns and adjectives exchangeable in the sentence and merging the two vocabularies V_{NN} and V_{SNei} into a united vocabulary V accordingly. The second model can be interpreted as running a unigram LDA on a non-discriminating mixture of nouns and adjectives.

To evaluate the general performance of our models, we conduct 46 independent runs of Gibbs sampling of 6000 iterations to train the three models to extract from variant number of clusters while ten percent sentences are left out for testing. All evaluation measures discussed on Section 4 are computed with samples drawn after 500 iterations of Gibbs sampling on the test set. Hype-parameter is set to be update every 20 Gibbs sampling iterations. The performance of perplexity results, average cluster entropies and normalized mutual entropy index measure are shown in Fig. 3, 4 and 5 respectively. In Fig. 3, lowest perplexity is achieved by SDWP on noun marginally which implies that SDWP is superior to the unigram LDA as SDWP combines naturally the two views of semantic dependency between nouns and neighbors and the association of nouns co-occurring in one sentence. The average perplexity of SDWP is also lower than the bag-of-word based LDA ("LDA on mixture") due to the effective introduction of dependency



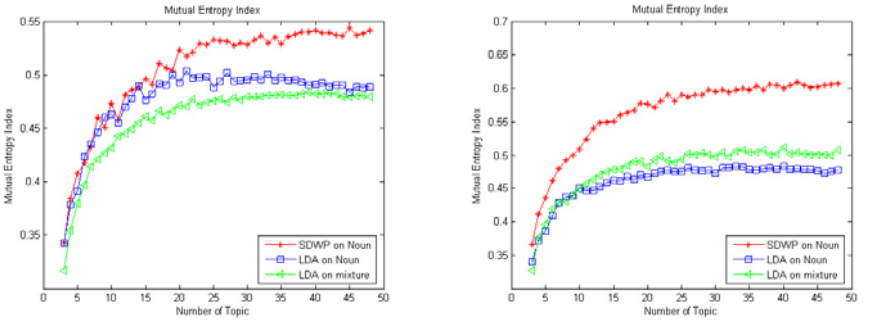
(a) log(Perplexity) of CitySearch dataset (b) log(Perplexity) of TripAdvisor dataset

Fig. 3. Logarithmic Perplexity on two datasets where "SDWP on noun" is perplexity on marginalized likelihood of noun and "SDWP on Pair" is perplexity of SDWP averaging between noun and adjective computed as Equation (11)



(a) ACE on CitySearch dataset (b) ACE of TripAdvisor Dataset

Fig. 4. Average cluster entropies of SDWP and the two baseline modes



(a) NMI of CitySearch Dataset (b) NMI of TripAdvisor Dataset

Fig. 5. Measure by normalized mutual entropy index on SDWP and the baseline models

Table 4. Top 10 items of each of the 15 noun clusters extracted by sentence based unigram LDA on the noun vocabulary in CitySearch dataset. Underlined terms indicate semantic relevance to the associated cluster.

1	steak, chicken, ribs, potatoes, shrimp, fries, tuna, spinach, salmon, sandwich
2	chocolate, desserts, cupcakes, gras, cookies, specials, cakes, crepes, pies, pastries
3	service, staff, atmosphere, decor, prices, ambiance, waitstaff, menu, experience, music
4	staff, atmosphere, drinks, crowd, music, bar, dining room, night, spot, bartenders
5	staff, dishes, wine list, menu, wines, wine, owners, tapas, price, bartender
6	tables, decor, space, interior, lighting, walls, bar, dining room, atmosphere, wood
7	pizza, bread, oil, crust, fare, course, portion, steaks, ingredients, fish
8	cheese, toast, flavor, burger, bread, style, dish, potato fries, salad, hot dog
9	chicken, menu, beef, sauce, meat, rice, rib, mussels, dumplings, noodles
10	service, waitress, waiter, experience, waiters, servers, hostess, server, night, wait staff
11	dishes, flavors, sushi, menu, cuisine, fish, service, rolls, chef, fare
12	coffee, tea, meats, bistro, ice cream, pie, zimbabwe, baristas juggle orders, home, cheeses
13	margaritas, dinner, brunch, cafes, afternoon, armchair, cream, guacamole, services, saturday night
14	service, restaurants, prices, spot, meals, party, addition, trip, patio, joint
15	appetizers, courses, entrees, chicken, dessert, salmon, potatoes, appetizer, course, steak

relation between noun and neighbors. It is a promising direction in future to extend the work to include all pairs of dependency relation. Lowest average cluster entropies attained by SDWP as shown in Fig. 4 indicates the clusters generated by SWDP is more cohesive and clusters give more probability mass to terms which is more discriminative and representative for the clusters, also evident from the extracted topic words in Table 5. The higher normalized mutual entropy index of SDWP in Fig. 5 is also another indication of the consistence and higher cohesion of the clusters of SDWP than that of the baseline models. All the three measures demonstrate consistently that our model SDWP outperforms the unigram LDA for the task of finger-grain product feature mining. Top 10 terms of each cluster extracted by SDWP on restaurant review dataset is shown in Table 5, which is obtained by ranking the terms with their cluster conditional

Table 5. Top 10 items of each cluster extracted by SDWP model from the CitySearch dataset. On the right side of each cluster index of the 15 clusters, upper row is noun fragments and down-side row is the associated group of semantic neighbor. Underlined terms indicate semantic relevance to the associated cluster.

1	tables, space, spot, seating, table, dining room, bar, dining, party, cafe
	outdoor, comfortable, empty, hot, main, cozy, outside, tiny, east, private
2	chicken, steak, tuna, salmon, duck, oil, shrimp, bass, appetizer, gras
	grilled, delicious, fried, perfect, amazing, dry, olive, rare, tasty, special
3	staff, waiter, waitstaff, waitress, wait staff, waiters, server, servers, bartender, bartenders
	friendly, attentive, helpful, knowledgeable, polite, pleasant, professional, courteous, knowledgable, cute
4	experience, meal, night, dinner, service, lunch, visit, spot, waiter, dining experience
	overall, real, late, perfect, special, romantic, amazing, entire, quiet, pleasant
5	atmosphere, decor, ambience, ambiance, space, crowd, service, setting, vibe, environment
	cozy, warm, romantic, cool, beautiful, perfect, comfortable, friendly, trendy, modern
6	desserts, wine, chocolate, drinks, dessert, cake, hot dogs, sangria, margaritas, brulee
	delicious, hot, tasty, white, perfect, sweet, amazing, complimentary, creme, rich
7	music, lighting, decor, interior, walls, wood, space, dining room, band, tables
	live, white, loud, warm, soft, low, cool, dark, beautiful, bright
8	potatoes, fries, toast, cheese, chicken, spinach, bread, burgers, potato fries, appetizers
	delicious, french, sweet, mashed, fried, grilled, cold, tasty, soft, blue
9	bread, crust, pizza, cheese, tea, pizzas, pasta, sauce, burger, sausage
	thin, delicious, hot, cold, sweet, iced, warm, amazing, thick, fantastic
10	dishes, menu, course, fish, sushi, courses, dish, appetizers, rolls, salads
	main, delicious, tasty, fixe, special, creative, unique, amazing, simple, raw
11	prices, wine list, drinks, service, menu, price, wines, wine, cocktails, desserts
	reasonable, delicious, amazing, extensive, affordable, worth, expensive, tasty, average, priced
12	ribs, rice, fish, meat, rib, beans, beef, chicken, pork, lamb
	delicious, fried, short, tender, black, flavorful, grilled, dry, tasty, sweet
13	service, ambience, services, ambiance, wine, seating, attitude, ambiance, meal, staffs
	friendly, attentive, prompt, efficient, slow, warm, professional, impeccable, quick, helpful
14	menu, cuisine, dishes, restaurants, fare, flavors, ingredients, cooking, wines, bistro
	italian, french, japanese, authentic, traditional, asian, american, mexican, indian, delicious
15	service, waiter, staff, hostess, waitress, manager, waiters, drinks, management, atmosphere
	rude, bad, slow, poor, terrible, horrible, worst, awful, obnoxious, cold

probability learned by the model. Similarly, top 10 terms of each cluster can be obtained by sentence based unigram LDA from nouns of CitySearch dataset. It can be found that the clusters extracted by SDWP are more descriptive and informative by both the groups of nouns and neighbors for each cluster. For example, it is clear to indicate product features as staff & service, grilled food and atmosphere in cluster 2, 3 and 5 in Table 5 respectively while the associated clusters in Table 4 are not so clear, e.g. cluster 3 is the mixture of staff and atmosphere. There are some interesting new clusters special in Table 5, such as

"internal environment" in cluster 7, "comparison of cuisine" in cluster 14 and etc. Furthermore, there are more descriptive and discriminative fine-grained clusters in Table 5, e.g. "good" and "bad" staff is differentiated between cluster 3 and 15, "internal" and "outdoor" environment differentiated between cluster 1 and 7 and etc. To analysis the disadvantage of conventional LDA, it is found that noise is brought in when conventional LDA takes into account the co-occurrence of nouns and adjectives without dependency relation.

Highly cohesive fine-grained product features are obtained as shown in Table 5. Furthermore, the associated group of semantic neighbors helps to discriminative the context of the features, e.g. cluster 3 and 15 in Table 5, which cannot be achieved by the conventional methods. The semantic neighbors of each SDWP cluster contain subjective and objective terms where objective terms provide fact context of each product feature and the subjective terms are good semantic candidates to build feature-specific opinionated words vocabulary for further sentiment analysis. The clusters generated by LDA on the non-discriminative mixture of noun and adjective are shown in Table 6, where the terms are not as informative and cohesive as that of SDWP in Table 5. Due to limitation of paper number,Comparison between results on TripAdvisor dataset obtained by SDWP and unigram LDA will not be shown in this paper. Interested readers could find full results on both dataset at "http://www.comp.hkbu.edu.hk/~tjzhan/SDWP_Results.html".

Table 6. Top 10 items of each of the 15 noun clusters extracted by unigram LDA on non-discriminative mixture of nouns and adjectives in CitySearch dataset. Underlined terms indicate semantic relevance to the associated cluster.

1	white, lighting, low, decor, walls, warm, dark, key, tables, wood
2	menu, french, dishes, italian, wine list, traditional, asian, wines, cuisine, fare
3	hot, cold, chocolate, spot, creme, tea, available, iced, fantastic, warm
4	service, rude, slow, bad, waiter, poor, staff, worst, experience, horrible
5	real, restaurants, italian, true, mexican, authentic, culinary, japanese, chef, cuisine
6	delicious, menu, special, drinks, dishes, tasty, sushi, amazing, creative, fish
7	atmosphere, decor, cozy, service, warm, ambience, beautiful, cool, romantic, ambiance
8	outdoor, seating, tables, space, night, bar, dinner, table, late, beautiful
9	friendly, service, staff, attentive, helpful, knowledgeable, waiter, prompt, professional, waitstaff
10	music, live, loud, crowd, cuban, mixed, cool, band, happy, jazz
11	prices, service, reasonable, experience, worth, price, overall, meal, decent, fine
12	grilled, delicious, chicken, fried, ribs, sweet, short, potatoes, black, tender
13	main, course, courses, dishes, dish, east, upper, dim, single, decent
14	delicious, fried, thin, bread, dry, cold, cheese, french, tasteless, perfect
15	delicious, sweet, perfect, desserts, wine, tasty, appetizers, dessert, amazing, margaritas

To measure the quality of representative terms for the clusters in a quantitative manner, a semantic relevance score is defined as ratio of number of manual evaluated semantic relevant items, namely, the underlined items, to the number

of items representing each cluster in the Table 4, 5 and 6. For example, the semantic relevance score of cluster 1 in Table 5 is 15/20. Comparison of averaged semantic relevance scores of clusters extracted by SDWP and the two baseline models is shown in Table 7, where the each cluster is represented by ten top items of noun and (or) adjective. The results in Table 7 are consisting with the topical results and discussion above, which demonstrate that SDWP outperforms the other two unigram LDA baseline models.

Table 7. Noun-neighbor pairs generation rules from dependency tree of a sentence

Model	SDWP	LDA on noun	LDA on mixture of noun and neighbor
Averaged SRS	0.87	0.76	0.80

7 Conclusion

Different from the bag-of-word approach in this paper, we model sentence as pairs of noun and adjective with semantic dependency. The semantic dependency between nouns and adjectives are combined with the sentential co-occurrence between nouns to develop a semantic dependent word pair generative model to extract semantic cohesive clusters of nouns and of adjectives for representing fine-grained product feature. Gibbs sampling is applied to infer the hidden variables and to infer the parameters. To evaluate the performance of the model, we computes the perplexity, average cluster entropies and normalize mutual entropy index on the samples drawn from the Gibbs sampling. The sentence based LDA is employed as the baseline mode represented by bag-of-noun corpus and non-discrimination mixture of nouns and adjectives corpus respectively. The experimental results demonstrate the advantage of our model and show promising direction on further research.

Acknowledgments. This work is partially supported by HKBU research grant FRG2/09-10/052.

References

1. Hu, M., Liu, B.: Mining and Summarizing Customer Reviews. In: ACM SIGKDD (2004)
2. Ding, X., Liu, B., Yu, P.S.: A Holistic Lexicon based Approach to Opinion Mining. In: WSDM, pp. 231–239 (2008)
3. Brody, S., Elhadad, N.: An Unsupervised Aspect-Sentiment Model for Online Reviews. In: NAACL, Los Angeles, CA, pp. 804–812 (2010)
4. Guo, H., Zhu, H., Guo, Z., Zhang, X.X., Su, Z.: Product feature categorization with multilevel latent semantic association. In: CIKM, pp. 108–121 (2009)
5. Raju, S., Shishtla, P., Varma, V.: A Graph Clustering Approach to Product Attribute Extraction. In: Indian International Conference on Artificial Intelligence (2009)

6. Joshi, M., Penstein-Ros, C.: Generalizing dependency features for opinion mining. In: ACL-IJCNLP (2009)
7. Zhan, T.J., Li, C.H.: Product Feature Mining with Nominal Semantic Structure. In: IEEE/WIC/ACM International Conference on Web Intelligence (2010)
8. Wallach, H.: Structured Topic Models for Language", PhD thesis, University of Cambridge (2008)
9. A NLP package, http://opennlp.sourceforge.net/
10. Blei, D.M., Ng, A.Y., Jordan, M.I., Lafferty, J.: Latent Dirichlet Allocation. JMLR (2003)
11. Nivre, J., Hall, J.: MaltParser: A Language-Independent System for Data-Driven Dependency Parsing. In: Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (2005)
12. Ganu, G., Elhadad, N., Marian, A.: Beyond the stars: Improving rating predictions using review text content. In: WebDB (2009)
13. TripAdvisor datasets discussed in this paper, http://patty.isti.cnr.it/~baccianella/reviewdata/
14. Wallach, H., Mimno, D., McCallum, A.: Rethinking LDA: Why Priors Matter. In: NIPS (2009)
15. A popular public stop-word list, http://truereader.com/manuals/onix/stopwords.html

Grammatical Dependency-Based Relations for Term Weighting in Text Classification

Dat Huynh, Dat Tran, Wanli Ma, and Dharmendra Sharma

Faculty of Information Sciences and Engineering
University of Canberra
ACT 2601, Australia
{dat.huynh,dat.tran,wanli.ma,
dharmendra.sharma}@canberra.edu.au

Abstract. Term frequency and term co-occurrence are currently used to estimate term weightings in a document. However these methods do not employ relations based on grammatical dependency among terms to measure dependency between word features. In this paper, we propose a new approach that employs grammatical relations to estimate weightings of terms in a text document and present how to apply the term weighting scheme to text classification. A graph model is used to encode the extracted relations. A graph centrality algorithm is then applied to calculate scores that represent significance values of the terms in the document context. Experiments performed on many corpora with SVM classifier show that the proposed term weighting approach outperforms those based on term frequency and term co-occurrence.

Keywords: Text representation, relation extraction, grammatical dependency, graph weighting model, text classification.

1 Introduction

In text classification, a single or multiple category labels will be automatically assigned to a new text document based on category models created after learning a set of labelled training text documents. Current text classification methods convert a text document into a relational tuple using the popular vector-space model to obtain a list of terms with corresponding frequencies.

Term frequency (TF) has been used to measure the importance levels of terms in a document. Firstly, TF is considered as a key component to evaluate term significances in a specific context [11]. The more a term is encountered in a certain context, the more it contributes to the meaning of the context. Secondly, some approaches have combined TF and Inverse Document Frequency (IDF) as a term weighting measure. These approaches outcome the considerable results as applied to text classification tasks [4,16,17]. However, with an abstract and complex corpus such as Ohsumed[1], the TF-based methods fail to leverage

[1] ftp://medir.ohsu.edu/pub/ohsumed

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 476–487, 2011.

the classification results [4,17]. According to Hassan and Banea [2], TF-based approaches can be effective for capturing the relevance of a term in a local context, but they fail to account for the global effects that terms exist on the entire document.

To overcome this shortcoming, the relationships among terms have been investigated to figure out the representation of a document. Recent studies have introduced the pre-defined relations among terms. These relations can be extracted from a predefined knowledge source such as Wikipedia Encyclopedia [1,3,12,14], in which the relations (Wikipedia links) are regarded as the main components to represent for the document context. In a case of finding out the methods that can extract the representation of unpredictable text documents, these pre-tagging-based methods are limited to encounter to the variety kinds of document.

The term co-occurrence (TCO) are popularly used to model the relations between terms [2,15]. It is also a model to get over the shortcoming of TF-based methods as well as to deal with the universal kinds of text documents. The idea of taking term co-occurrence as a relation is not only to capture the dependency of terms in local contexts but also to take into account the global effects of terms in the entire document. In order to estimate importance levels of terms, a graph model is used to connect all these relations and a centrality algorithm is used to calculate term weighting values.

Although TCO-based methods give the considerable results in comparison to the TF-based methods when they are used to estimate important terms of a given document for TC tasks [2], some certain concerns need to be considered. Firstly, when working in a certain window size as the local context, these methods accept every single pair of terms within the window size as a relation. So, the number of relations would be small or even really large depending on the choice of the window size. In those cases, the expectable relations can be eliminated, or the redundancy relations are still retained. Secondly, although the idea of these approaches is to extract important terms in a document context, the way of making pair of relations under a window size does not guaranty for the contribution of the relations in weighting important terms. With those explanations, we argue that not only TF-based model but also TCO-based model may not be the best technique to capture those important terms.

In this paper, we propose an alternative method to extract and weighting important terms in a given document. The method firstly considers the advantages of the relations among terms to address the shortcoming of TF-based methods. Secondly, under the light of the success of TextRank [9,2], instead of using term co-occurrence as a dependency, we are more concentrating on relations based on grammatical dependency among terms. The choice of the relations not only prevents the issues of window sizes but also discloses more hidden relations as walking along the path connected terms.

As the framework of our method, we start firstly with extracting relations from the given text document. Secondly, the graph model is used to encode contexts of the document, which is constructed by connecting all these relations.

Then, a graph centrality algorithm is applied to calculate scores that represent significant values of terms in the document context. Finally, the top list of high weighted terms will be used as a representation of the given document.

The remaining of this paper is organised as follows. In Section 2, a framework of term weighting approach is presented. Section 3 explains the methodology of extracting grammatical relationships among words. Section 4 shows how to use the graph model to estimate the importance level of terms. How to apply the document representation to text categorisation tasks is presented in Section 5. Section 6 describes experimental setups, results and discussions. Finally, a conclusion and future work will be discussed in Section 7.

2 The Proposed Term Weighting Framework

The term weighting framework includes the following phases (see Fig. 1):

1. *Relation Extraction:* Given an input document, the relation extraction phase extracts a set of tuples (relations) representing to the document.
2. *Term Weighting:* A weighted graph is conducted by encoding all the extracted relations. A graph ranking model is used to estimate term weightings for document representation.

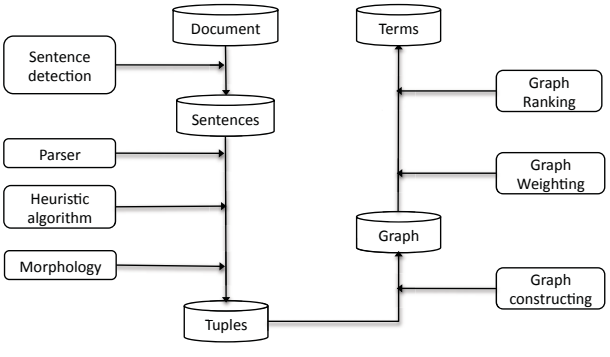


Fig. 1. The proposed term weighting framework

The proposed framework is capable of extracting more relationships among terms within a document, and the relations include not only grammatical relations, but also hidden relations which are explored by walking along the paths connecting the ideas of each sentence. A global context of the document is captured using a graph model and a centrality ranking algorithm. The graph model is able to inter-connect the major ideas of the document together, and is a place for the centrality algorithm to estimate the important levels of vertices. The term weighting framework allows the important terms of a document to be "voted" by other terms in the same document.

3 Relation Extraction

Relation extraction is an important research area in text mining, which aims to extract relationships between entities from text documents. In the framework, a relation is considered as a tuple $t = (e_i, r_{ij}, e_j)$, where e_i and e_j are strings denoted as words (terms), and r_{ij} is a string denoted as the relationship between them.

The relation can be extracted based on linguistic analysis, particularly grammatical relations among terms are the key components for extracting information.

For instance, from the following sentence *"Antibiotics kill bacteria and are helpful in treating infections caused by these organisms"*, a list of relations extracted includes *(Antibiotic, kill, bacteria)*, *(Antibiotic, treat, infection)*, *(Antibiotic, treat infection cause, organism)*, and *(infection, cause, organism)*.

In order to extract relations, the sentences are identified from the input text document using a sentence detection technique, a linguistic parser such as Stanford parser[2] is then used to analyse each sentence and outputs a graph of linkages (Fig. 2), and finally a heuristic algorithm is designed to walk along paths from the linkage graph and extract the expectable relations.

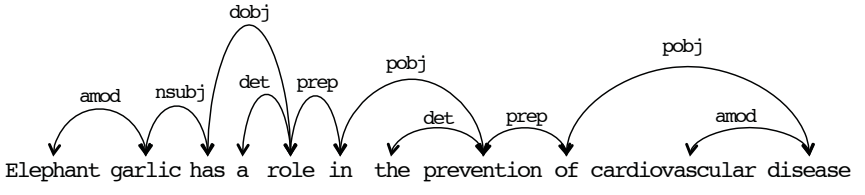


Fig. 2. Graph of linkages extracted from the following sentence: *"Elephant garlic has a role in the prevention of cardiovascular disease"*. Each label assigned each curve plays a role of a grammatical relation connecting two words. An expectable relation can be extracted based on the grammatical relation only or based on the shortest paths connecting terminated words.

The heuristic algorithm firstly scans the parsed sentence and identifies pairs of base terms[3] (e_i, e_j) with $i < j$. From each pair of base terms, if there is a shortest path connecting from e_i to e_j, the algorithm will go along the path to identify a sequence of words between e_i and e_j. These order words is considered as a potential connection r_{ij} to form the raw tuple $t = (e_i, r_{ij}, e_j)$. If e_i and e_j are connected directly, the connection r_{ij} is regarded as the name of the linkage (label). Finally, if the raw tuples are passed through all given constraints, they will be retained for the next processing step. Constraints to test the raw tuples include:

[2] The Stanford parser http://nlp.stanford.edu/software/lex-parser.shtml

[3] In this case, a base term is a single word in accordance with POS filter.

- e_i and e_j have to be base terms with POS filter
- r_{ij} has to be in a shortest path connecting e_i to e_j
- r_{ij} has to contain a verb or a preposition, or it is a grammar connection

After extracting the set of raw tuples from each document, components in each tuple should be optimised as follows. All non-essential words such as adverbs, relative clause marker (who, whom, which, that, etc.), and stop-words will be eliminated from all components of tuple. The morphology technique is then used to convert all words to their simple forms, such as converting plural words into singular form and any kinds of verb forms into their *"root"* words [8]. For instance, the noun phrase *"developing countries"* is converted to *"develop country"*, and the verb phrase *"have been working"* is converted to *"have be work"*. Once all wired tuples are eliminated, the remaining tuples are considered as a set of relations represented the document and is ready to build the graph for selecting term representatives.

4 Graph Construction: Constructing, Weighting and Ranking Graph

Graph model is an alternative way to model information to show relationships between vertices. It groups related information in a certain way that the centrality algorithms can take the best advantages.

4.1 Constructing Graph

A graph model is built to connect all extracted relations. Given a relation $t = (e_i, r_{ij}, e_j)$, where e_i and e_j are considered as vertices in the graph and r_{ij} is considered as an edge connecting between e_i and e_j. The weighting of the edge r_{ij} is calculated based on the redundancy of the tuple t and the relatedness between e_i and e_i.

4.2 Weighting Graph

The weighting of a edge $w(r_{ij})$ is calculated based on two factors. Firstly, it depends on the frequency of the relation t in the document d. The higher redundancy of relation t is, the more important it is in the document d. Secondly, the weighting of the edge $w(r_{ij})$ is also based on the redundancy of relation t in the corpus. The redundancy of a tuple determines how valuable of that information from its document [6].

Let $t = (e_i, r_{ij}, e_j)$ be a relation of d, and $e = (e_i, w(r_{ij}), e_j)$, be an edge of the graph, the weighting $w(r_{ij})$ is calculated as:

$$w(r_{ij}) = freq(t, C) * rf(t, d) \tag{1}$$

$$rf(t, d) = \frac{freq(t, d)}{\sum_{i=1}^{|t:t \in d|} freq(t_i, d)} \tag{2}$$

where $freq(t, C)$ is the frequency of tuple t in the corpus C, $freq(t, d)$ is the frequency of tuple t in the document d, and $rf(t, d)$ is the relation frequency value of the relation t in the document d.

4.3 Ranking Graph

Once a document is represented as a weighted graph, a graph ranking algorithm is used to estimate scores of its vertices. According to Sinha and Mihalcea [10], the centrality algorithm PageRank [5] shows its outstanding ability on weighting graph and hence it is adapted in our approach. The basic idea of PageRank algorithm is that a web page will have a high rank if there are many web pages or high ranking web pages pointing to it. Therefore, we treat each node in the term graph as a web page, every undirected edge $e = (e_i, w(r_{ij}), e_j)$ needs to be converted to two directed edges $\overrightarrow{e} = (e_i, w(r_{ij}), e_j)$ and $\overleftarrow{e} = (e_j, w(r_{ij}), e_i)$. Then the directed graph is passed through the PageRank as its input and the output is a set of vertices with their ranking scores. Every vertex (term) e_i in the graph (document) d has its ranking score $pr(e_i, d)$, which is considered as degree of significance of the vertex (term) in the graph (document).

5 Applying Graph-Based Document Representation to Text Classification

Given a text document d from a corpus C, the list of n term representatives of d is defined as

$$d = \left\{ \Big(w_1, pr(w_1, d)\Big), \Big(w_2, pr(w_2, d)\Big), \ldots, \Big(w_n, pr(w_n, d)\Big) \right\} \tag{3}$$

where w_i is the text value of term i in the document d and $pr(w_i, d)$ is the ranking value of term w_i of the document d. The list of categories of the corpus C is $C = \{c_1, c_2, \ldots, c_m\}$.

In the following section, we proposes a measure that takes into account dependencies between terms and classes, which can justify term weighting values to adapt with text classification tasks.

5.1 Proposed Term Class Dependence (TCD)

The idea of calculating class dependence value of terms is that terms represented for a document are normally dependent of its categories (classes). Therefore, we propose a measurement *tcd*, which takes into account the information of categories and denotes the degree of dependence of a term to a particular category. If a word frequently occurs in many documents of one class and infrequently occurs in other classes, it should be considered as representative for the class if its ranking value from the document is also comparable. We suggest a measure of degree of dependence of the term w_i to the category c_i

$$tcd(w_i, c_j) = \frac{tf(w_i, c_j) * df(w_i, c_j)}{\sum_{k=1}^{m} tf(w_i, c_k) * df(w_i, c_k)} \quad (4)$$

where c_k is a categories of the corpus C.

With the purpose of classifying text, we propose a combination of term ranking on documents *(pr)* and category-based calculation *(tcd)*, which establishes a formula to weight terms w_i from a document d_j that belongs to a category c_k as follows:

$$pr.tcd(w_i, d_j, c_k) = pr(w_i, d_j) * tcd(w_i, c_k) \quad (5)$$

and the *pr.tcd* value of each term will be added to the feature vector for classification task.

5.2 Proposed Hybrid Term Weighting Methods Based on TCD

With the purpose of evaluating the effectiveness of term weighting methods based on term frequency, term co-occurrence relations, and grammatical relations, we suggest the following combinations to form hybrid term weighting methods for text classification:

- ***tf.tcd***: a combination of term frequency and term class dependency. The purpose of *tf.tcd* is to evaluate the effectiveness between *tf* and *pr* when making the comparison between *tf.tcd* and *pr.tcd*. The *tf.tcd* value of a term w_i in a document d_j is

$$tf.tcd(w_i, d_j) = tf(w_i, d_j) * tcd(w_i, d_j) \quad (6)$$

where

$$tf(w_i, d_j) = \frac{freq(w_i, d_j)}{\sum_{k=1}^{n} freq(w_k, d_j)} \quad (7)$$

n is number of terms in the document d_j.

- ***rw.tcd***: a combination term weighting method based TextRank *(rw)*[4] and term class dependency (tcd). The purpose of *rw.tcd* is to evaluate the effectiveness between *rw* and *pr* when making the comparison between *rw.tcd* and *pr.tcd*. The *rw.tcd* value of a term w_i in a document d_j is

$$rw.tcd(w_i, d_j) = rw(w_i, d_j) * tcd(w_i, d_j) \quad (8)$$

where $rw(w_i, d_j)$ is the random-walk weighting value of a term w_i in a document d_j.

[4] We implemented the TextRank framework which outcomes the weightings of terms based on term co-occurrence and random walk algorithm on the graph [9,2] for comparison purposes.

– ***pr.idf***: a combination our term ranking method *pr* and invert document frequency. The purpose of *pr.idf* is to evaluate the effectiveness between *tf* and *pr* when making the comparison between *tf.idf* and *pr.idf*. The *pr.idf* value of a term w_i in a document d_j is

$$pr.idf(w_i, d_j) = pr(w_i, d_j) * idf(w_i, d_j) \tag{9}$$

6 Experiments

6.1 Classifier and Data Sets

Support Vector Machine [13] is a state-of-the-art classifier. In our experiments, we used the linear kernel since it was proved to be as powerful as other kernels when tested on data sets for text classification [16].

Our term weighting approach is mainly based on the grammatical relations among terms in English sentences. In order to test its effectiveness in comparison to other methods, we have chosen highly standard English grammar corpora which are Wikipedia XML, Ohsumed, and NSFAwards.

Wikipedia Corpus: We used the collection Wikipedia XML Corpus compiled by Denoyer & Gallinari [7]. We randomly selected a subset of the English Single-Label Categorization Collection which provides a single category to each document. After the pre-processing step, we obtained a total of 8502 documents assigned to 53 categories. These documents are divided randomly and equally to form a training data set and a test data set including 4251 documents and 53 categories for each set.

Ohsumed: This corpus contains 50216 abstract documents[5], we selected the first 10000 for training and the second 10000 for testing. The classification task is to assign the documents to one or multiple categories of the 23 MeSH "diseases" categories. After pre-processing step, there are 7643 documents in the test set and 6286 documents in the training set.

NSFAwards: This data set consists of 129000 relatively short abstracts in English, describing awards granted for basic research by the US National Science Foundation during the period 1990-2003. For each abstract, there is a considerable amount of meta-data available, including the abbreviation code of the NSF division that processed and granted the award in question. We used this NSF division code as the class of each document. The title and the content of the abstract were used as the main content of the document for classification tasks. We used part of the corpus for our experiment by selecting 100 different documents for each single category, test set and training set. After the pre-processing step, we obtained 19018 documents for training set, 19072 documents for test set and 199 categories.

[5] ftp://medir.ohsu.edu/pub/ohsumed

6.2 Performance and Discussion

To evaluate the classification system we used the traditional accuracy measure defined as the number of correct predictions divided by the number of evaluated examples. Six weighting models that need to be tested are $tf.idf$, $tf.tcd$, $rw.idf$, $rw.tcd$, $pr.idf$ and $pr.tcd$. We used the accuracy results from $tf.idf$, and $rw.idf$ as the baseline measures.

GR-based Method versus Baseline Methods. The GR-based method $pr.tcd$ provided outstanding results in comparison to $tf.idf$ and $rw.idf$. Table 1 shows the classification results using the SVM classifier. The Ohsumed corpus is one of the challenging text classification datasets when $tf.idf$ and $rw.idf$ achieved under 40% accuracy. However, $pr.tcd$ achieved a considerable result with about 16% accuracy higher than the two previous methods. Wikipedia and NSFAwards also show the similarity trends, yet the gap between $pr.tcd$ and two baseline methods reduced to about 4% from NSFAwards corpus and to about 10% from Wikipedia corpus.

Table 1. SVM results among of $pr.tcd$, $tf.idf$ and $rw.idf$ weighting methods

	$tf.idf$	$rw.idf$	$pr.tcd$
$Wikipedia$	73.27%	74.99%	**84.92**%
Ohsumed	39.75%	39.02%	**56.9**%
$NSFAwards$	67%	64.98%	**70.9**%

Moreover, from the classification results of six weighting methods in Table 2, it is seen that the GR-based method *(pr.tcd)* also shows the comparable results to other similarity methods.

Table 2. SVM results from six weighting schemas

	$tf.idf$	$tf.tcd$	$rw.idf$	$rw.tcd$	$pr.idf$	$pr.tcd$
$Wikipedia$	73.27%	77.4%	74.99%	82.66%	77.72%	**84.92%**
Ohsumed	39.75%	39.73%	39.02%	56.03%	39.24%	**56.9%**
$NSFAwards$	67%	**71.2%**	64.98%	70.95%	65.2%	70.9%

Grammatical Relation versus Term Frequency: The effectiveness of grammatical relation and term frequency can be measured when making comparisons of the accuracy results of two pairs $(tf.idf, pr.idf)$ and $(tf.tcd, pr.tcd)$. The chart from Fig. 3 shows that pr always gives better performance than tf when it is combined with tcd in the term weighting methods. However, this trend is not stable when pr goes along with idf. Particularly, $pr.idf$ presents outstanding performance on Wikipedia corpus, but $tf.idf$ shows its strength on the other two corpora Ohsumed and NSFAwards.

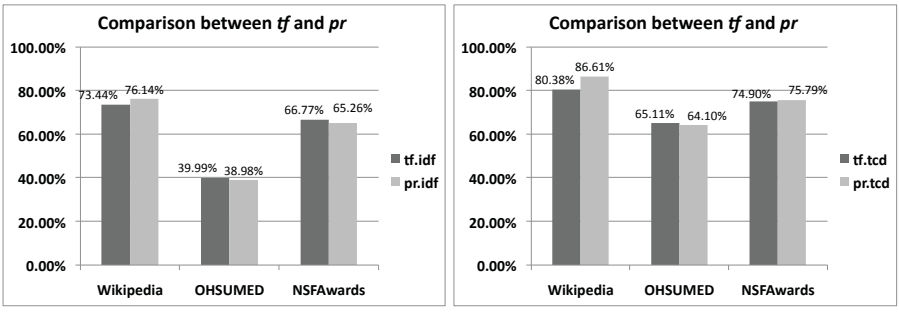


Fig. 3. The chart shows the accuracy comparison between term weighting schema based on term frequency and grammatical relation

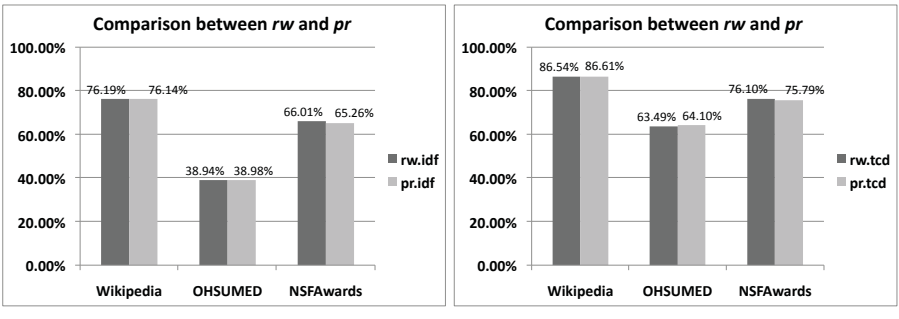


Fig. 4. The chart shows the accuracy comparison between term weighting schema based on term frequency and grammatical relation

Grammatical Relation versus Term Co-occurrence Relation. The ideas behind term weighting schema based on these methods have some similarity. However, each of them has their strengths and weaknesses. The effectiveness of these approaches can be measured by taking the comparison between two pairs of weighting schemas $(rw.idf, pr.idf)$ and $(rw.tcd, pr.tcd)$. The chart from the figure 4 shows that the majority cases term weighting methods based on grammatical relations outperforms to those based on term co-occurrence relations. Particularly, the biggest gaps between the classification accuracy between two methods is 2.7%, whereas just only 1 out of 6 cases the TCO-based methods show the comparable result to GR-based methods. In the case of NSFAwards corpus, TCO-based methods achieve higher 0.1% accuracy results in comparison to GR-based methods.

Term Class Dependency versus Inverse Document Frequency. The information from the chart of Fig. 5 shows another view of information. It presents the comparison between the contribution of inverse document frequency and term class dependency measures in term weighting schemas. Most of the cases, whenever term important evaluation *(tf, rw, pr)* combines with *tcd* shows the

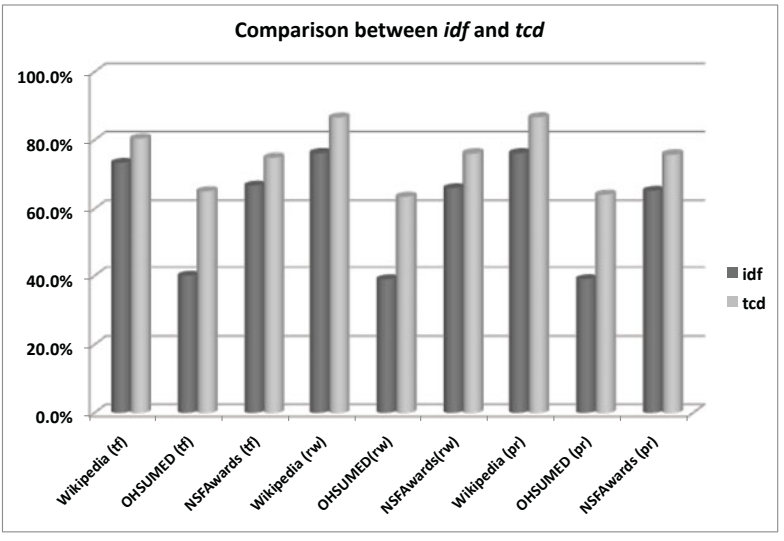


Fig. 5. The chart shows the accuracy comparison between term weighting schema based on term frequency and grammatical relation

outstanding results in compare with *idf*. There is just only one cases from the Ohsumed corpus that *idf* shows better results than *tcd*.

In the summary, we have presented the experiment results and have made the comparisons related to the strengths and weaknesses of proposed methods. Although some aspects need to be considered, the proposed term weighing approach for text classification using grammatical relations outperforms to other traditional term weighing approaches based on term frequency and term co-occurrence, and the term class dependency measure can be used as the alternative information evaluation instead of inverse document frequency.

7 Conclusion and Future Work

The paper has presented term weighting method for text classification based on grammatical relations. With the same datasets, the approach has improved accuracy of text classification in comparison to the traditional term weighting method. The approach overcomes the less of frequency of information by self-creating the frequency based on the grammar structure of text content. This approach also raises motivations for our further investigation on the benefits of relations on text classification as well as text mining.

Our approach uses the concept of relations, we still do not take the closed considerations on its semantic aspect, we have just used the relations as connections between terms as a first attempt for getting more statistical information. For further investigation, we are more focusing on taking the semantic information from tuples and its connection from the graph to form representations

of given documents. Moreover, the grammatical relations is extracted based on the grammar structure of text body, this procedure has consumed much computational processing. Therefore, the need for quick and reliable extraction from input text should be considered as the further investigations.

References

1. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proc. of the 20th IJCAI, pp. 1606–1611 (2007)
2. Hassan, S., Banea, C.: Random-walk term weighting for improved text classification. In: Proc. of TextGraphs, pp. 53–60 (2006)
3. Hu, J., Fang, L., Cao, Y., Zeng, H.J., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging wikipedia semantics. In: Proc. of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 179–186 (2008)
4. Joachims, T.: Text categorisation with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
5. Lawrence, P., Sergey, B., Rajeev, M., Terry, W.: The pagerank citation ranking: Bringing order to the web. In: Stanford Digital Library Technologies Project (1998)
6. Ludovic, D., Patrick, G.: A probabilistic model of redundancy in information extraction. In: Proc. of the 19th IJCAI, pp. 1034–1041 (2005)
7. Ludovic, D., Patrick, G.: The wikipedia xml corpus. In: ACM SIGIR Forum, pp. 64–69 (2006)
8. Minnen, G., Carroll, J., Pearce, D.: Morphological processing of english. In: Natural Language Engineering, pp. 207–223 (2001)
9. Rada, M., Paul, T.: Textrank: Bringing order into texts. In: Proc. of the EMNLP (2004)
10. Ravi, S., Rada, M.: Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity. In: Proc. of the ICSC, pp. 363–369 (2007)
11. Robertson, S., Jones, K.S.: Simple, proven approaches to text retrieval. Tech. rep., University of Cambridge (1997)
12. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: Proc. of the 21st AAAI, pp. 1419–1424 (2006)
13. Vapnik, V.N.: The nature of statistical learning theory. Springer, Heidelberg (1995)
14. Wang, P., Hu, J., Zeng, H.J., Chen, L., Chen, Z.: Improving text classification by using encyclopaedia knowledge. In: The Seventh IEEE ICDM, pp. 332–341 (2007)
15. Wang, W., Do, D.B., Lin, X.: Term graph model for text classification. In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS (LNAI), vol. 3584, pp. 19–30. Springer, Heidelberg (2005)
16. Yang, Y., Liu, X.: A re-examination of text categorisation methods. In: Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 42–49 (1999)
17. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorisation. In: Proc. of the 14th ICML, pp. 412–420 (1997)

XML Documents Clustering Using a Tensor Space Model

Sangeetha Kutty, Richi Nayak, and Yuefeng Li

Faculty of Science and Technology
Queensland University of Technology
GPO Box 2434, Brisbane Qld 4001, Australia
{s.kutty,r.nayak,y2.li}@qut.edu.au

Abstract. The traditional Vector Space Model (VSM) is not able to represent both the structure and the content of XML documents. This paper introduces a novel method of representing XML documents in a Tensor Space Model (TSM) and then utilizing it for clustering. Empirical analysis shows that the proposed method is scalable for large-sized datasets; as well, the factorized matrices produced from the proposed method help to improve the quality of clusters through the enriched document representation of both structure and content information.

1 Introduction

Rapid growth of web technologies has witnessed a sudden surge in the number of XML (eXtensible Markup Language) documents. For instance, English Wikipedia contains 3.1 million web documents in XML format; the ClueWeb dataset, used in Text Retrieval Conference (TREC) tracks, contains 503.9 million XML documents collected from the web in January and February 2009. The majority of existing XML document clustering methods utilize either the structure features [2] or the content features present in the documents. Clustering methods utilizing only the content features of the documents consider the documents as a "bag of words" or a Vector Space Model (VSM) and ignore the structure features [2]; clustering methods utilizing only the structure features of the documents represent each document as a set of paths (sequences) or trees.

However, these methods, with their single-feature focus, tend to falsely group documents that are similar in for documents that are similar in both features. To correctly identify similarity among documents, the clustering process should use both their structure and their content information. Approaches on clustering both the structure and the content features of the XML documents are limited. Approaches using the VSM often fail to scale for even small collections of a few hundred documents, and in some situations have resulted in poor accuracy [14]. VSM cannot model both structure and content features of XML documents effectively as the mapping between the structure and its corresponding content is lost. The content and structure features inherent in an XML document should be modeled in a way that the mapping between the content of the path or tree

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 488–499, 2011.

can be preserved and used in further analysis. In this paper we propose a novel method that represents the XML documents in a Tensor Space Model (TSM) and uses the TSM for clustering. In the TSM, storing the content corresponding to its structure helps to analyze the relationship between structure and content.

Unlike the VSM, which uses a vector to model, TSM is based on the multilinear algebraic character level high-order tensors (generalization of matrices) [13]. Decomposition algorithms are used to analyze the relationships between various tensor orders (ways or modes). However, existing decomposition algorithms could be used to analyze small size and sparse TSMs. TSMs that are large and dense cannot be loaded into memory. Consequently, large datasets with tensor representation cannot be analyzed using these decomposition techniques. In this paper, we propose a randomized tensor decomposition technique that could upload the large size tensors into memory and decompose them with significant speedups. Experiments on a real-life dataset containing more than 50K documents show that the proposed method helps to improve the cluster quality through the enriched document representation of both structure and content information. The contributions of this paper can be summarized as: (1) a clustering method, XML document Clustering with TSM (XCT), that utilizes the tensor model to efficiently combine the content and structure features of XML documents; and (2) a new tensor decomposition algorithm, Progressive Tensor Creation and Decomposition (PTCD), for large sized tensors.

2 Related Work

Tensor Space Modeling (TSM) has been successfully used in representing and analyzing multi-way data in signal processing, web mining and many other fields [13]. Tensor clustering is a multi-way data analysis task which is currently gaining importance in the data mining community. The simplest tensor clustering scenario, co-clustering or bi-clustering, in which two orders are simultaneously clustered, is well established [6]. Another recently proposed approximation based Combination Tensor Clustering algorithm [7] clusters along each of the orders and then represents the cluster centers in the tensor. These co-clustering techniques capture only the 2-way relationships among the features and ignore the dependence of multiple orders in clustering: this may result in loss of information while grouping the objects.

Several decomposition algorithms, such as Higher Order SVD (HOSVD); CP, a higher-order analogue of Singular Value Decomposition (SVD) or Principal Component Analysis (PCA); Tucker and Multi-Slice Projection, have been reviewed in detail in [5]. Incremental Tensor Analysis (ITA) methods [8] have been proposed recently to detail with large datasets for efficiently decomposing sparse tensors ($density \leq 0.001\%$). However, real-life XML documents represented in TSM are dense with about 127M non-entries with over 1M terms and these decomposition algorithms fail to scale. MET [9], a memory-efficient implementation of Tucker proposed to avoid the intermediate blow-up in tensor factorization, is shown in our results shows not to scale to our medium-sized and large-sized

datasets. In MACH [13], a recently proposed random decomposition algorithm suitable for large dense datasets, the number of entries in the tensor is randomly reduced using Achlioptas-McSherry's technique [1] to decrease the density of the dataset. However, as discussed in section 5, MACH often ignores smaller length documents and tends to group most of the smaller length documents in a single cluster in spite of differences in their structure and content. To remove this lack of decomposition algorithms suitable for very large-sized datasets, in this paper we propose a new decomposition algorithm, the Progressive Tensor Creation and Decomposition (PTCD) algorithm, that progressively unfolds a tensor into a matrix and applies SVD on this generated matrix.

3 The Proposed XCT Method

3.1 Problem Definition and Preliminaries

Let there be a collection of XML documents $D = \{D_1, D_2, \ldots, D_n\}$, where D_i is an XML document containing tags and data enclosed within those tags. The structure of D_i can be defined as a list of tags showing the hierarchical relationships between them. The structure of D_i is modeled as a rooted, ordered and node-labeled document tree, $DT_i = (N, n0, E, f)$, where (1) N is the set of nodes that correspond to tags in D_i, with the node labels corresponding to tag names; (2) $n0$ is the root node which does not have any edges entering in it; (3) E is the set of edges in DT_i; and (4) f is a mapping function $f : E \rightarrow N \times N$. Previous research has shown that, in a dataset, only the content constrained within the concise common or frequent subtrees (Closed Frequent Induced - CFI) can be used to group the documents, rather than the entire content of the XML documents [10]. Therefore the proposed XCT method generates these CFI subtrees to represent the common subtrees in the dataset and uses these CFI subtrees to extract the content of the documents corresponding to them. The process begins by identifying the subtrees that belongs to a document tree. A subtree $CFI_j \in CFI$ is present in document tree DT_i, if CFI_j preserves the same parent-child relationship as that of DT_i. The document content(or *structure-constrained content*) contained within the CFI_j subtree in DT_i, noted as $C(D_i, CFI_j)$, is retrieved from the XML document D_i, a collection of node values or terms. The node value of a node (or tag) of a CFI_j, $C(N_i)$ in D_i is a vector of terms, $\{t_1, \ldots, t_k\}$ that the node contains. The term t is obtained after stop-word removal and stemming.

The next step involves modeling the derived structure and content features of a tensor model. Firstly,the tensor notations and conventions used in this paper are akin to the notations used by previous works [5,8,13]. Let $\mathcal{T} \in \mathbb{R}^{M_1 \times M_2 \times M_3 \times \ldots \times M_n}$ be a tensor of n orders where M_i is an order. In this work, we focus on the third-order tensor, $\mathcal{T} \in \mathbb{R}^{M_1 \times M_2 \times M_3}$. Entries of a tensor are shown using a_{ijk} and the subscript (i, j, k) range from I, J, K in each order. Each element (or entry) of a tensor needs n indices to represent or reference its precise position in a tensor. For example, the element a_{ijk} is an entry value at the i, j and k orders. Given the documents set D, its corresponding set of CFI subtrees and the set of terms

for each CFI subtree, the collection of XML documents is now represented as a third-order tensor $\mathcal{T} \in \mathbb{R}^{D \times CFI \times Terms}$. The tensor is populated with the number of occurrences of the structure-constrained term $Terms_i$ that corresponds to the CFI_j for document D_k. Two optimization techniques are applied on the two orders, CFI and $Terms$, to reduce the size of the tensor. Fig. 1 provides an overview of the XCT method. It begins with mining the CFI subtrees using the PCITMinerConst algorithm and then identifying the constrained content within those CFI subtrees for a given document. Once the structure and content features are obtained for each document, the documents are represented in the TSM along with their structure and content features. The next task is to decompose the created TSM to obtain factorized matrices. Lastly, the K-means algorithm is applied to one of the factorized matrices representing the left singular matrix for the "Document" order U_D and the clusters of documents are obtained.

Input: *Document Dataset;* ***D****, Document Tree Dataset:****DT****, Minimum Support:****min_supp****, Length Constraint:* ***len****, NumCluster:* ***c****, RI Vectors Length:* γ
Output: *Clusters:* $\{Clust_1 \ldots Clust_c\}$
Method:
1. Compute $CFI = \{CFI_1, \ldots, CFI_p\}$ for DT using the PCITMinerConst algorithm for the given *min_supp* and *len*.
2. Form clusters of similar CFI subtrees, $CFISC = \{(CFI_1, \ldots, CFI_q), \ldots, (CFI_t, \ldots, CFI_u)\}$, where $CFISC = \{CFISC_1, \ldots, CFISC_h\}$, $k \ll p$ using large itemset algorithm.
3. For every document $D_i \in D$
   a.Identify the $CFISC$ existing in DT_i, $\delta(DT_i) = \{CFISC_l, \ldots, CFISC_h\}$
   b. For every $CFISC_j$ in $\delta(DT_i)$ retrieve the structure-constrained content in D_i, $C(D_i, CFISC_j) = C(N_1), \ldots, C(N_m)$. The set $C(N_m) = t_1, \ldots, t_k \in Terms$, where $Terms$ is the term list in D.
4. Apply random indexing using the γ length random vectors on the terms collection to reduce the term space to $Terms'$
5. Form a tensor $\mathcal{T} \in \mathbb{R}^{D \times CFISC \times Terms'}$, where each tensor element is the number of times a term t_k occurs in $CFISC_j$ for a given document D_i.
6. Apply the proposed tensor decomposition algorithm, PTCD to the tensor $\mathcal{T}$ and get the resulting left singular matrices U_D, U_{CFISC} and $U_{Terms'}$.
7. Apply K-means clustering to U_D to generate the c number of clusters.

Fig. 1. High-level definition of XCT

3.2 Generation of Structure Features for TSM

The Prefix-based Closed Induced Tree Miner (PCITMiner) algorithm [10] is modified to generate the length-constrained CFI subtrees from the document tree dataset DT. The length constrained CFI subtrees are used in this method for the following reasons: (1) Extracting all the CFI subtrees is computationally expensive for datasets with a high branching factor; (2) All CFI subtrees are not required while utilizing them in retrieving the content. In fact the long sized CFI subtrees become more specific and result in retrieving distinct terms associated only with this tree. This may result in a higher number of clusters with uneven sizes. We call the modified algorithm the PCITMinerConst algorithm.

Fig. 1 illustrates the computationally expensive operation of checking whether the mined CFI exists in a given document tree due to the graph isomorphism problem. This step can be optimized by grouping similar subtrees based on their

similarity and then retrieving the content corresponding only to the group of similar CFI. A large itemset algorithm for clustering transactional data has been modified to include subtrees, rather than items, to conduct the grouping of the CFI trees based on the similarity of the subtrees. The clusters of CFI subtrees, called Closed Frequent Induced Subtree Cluster ($CFISC$), become a tensor order for representing and analyzing XML documents. Let $CFISC$ be a set of CFI subtrees given by $\{(CFI_1, \ldots, CFI_q)(CFI_r, \ldots, CFI_s)(CFI_t, \ldots, CFI_u)\}$.

3.3 Generation of Content Features for TSM

$CFISC$ is used to retrieve the structure-constrained content from the XML documents. We now define the coverage of a $CFISC_j$ and its constrained content for the given document D_i. Compared with the content features of an XML document, the structure-constrained content features include the node values corresponding only to the node labels of the set of CFI subtrees in $CFISC_j$.

Definition 1: Structure-Constrained Content Features. These features of a given $CFISC_j, C(D_i, CFISC_j)$ of an XML document D_i, are a collection of node values corresponding to the node labels in the $CFISC_j$ where $CFISC_j$ is a cluster of CFI subtrees corresponding to DT_i. The node value of a node (or tag) of a $CFISC_j \in CFISC, C(N_i)$, in D_i is a vector of terms, $\{t_1, \ldots, t_k\}$ that the node contains. The term t is obtained after using pre-processing techniques such as stop-word removal and stemming. Firstly, the CFI subtrees corresponding to the $CFISC_j = \{CFI_r, \ldots, CFI_s\}$ for a given document D_i are flattened into their nodes $\{N_1, \ldots, N_m\} \in N$, where N is the list of nodes in DT. Then the node values of $\{N_1, \ldots, N_m\}$ are accumulated and their occurrences for a document D_i are recorded.

In large datasets, the number of terms in the structure-constrained content is very large with more than 1M terms and 127M tensor entries for INEX (Initiative for Evaluation of XML retrieval) 2009 Wikipedia even after pre-processing. To reduce this very large term space, we apply a Random Indexing (RI) technique which has been favored by many researchers due to its simplicity and low computationally complexity [12]. In RI, each term in the original space is given a randomly generated index vector as shown in Fig. 2. These index vectors are sparse in nature and have ternary values (0 , -1 and 1). Sparsity of the index vectors is controlled via a seed length that specifies the number of randomly selected non-zero features.

$$r_{ij} = \sqrt{3} \begin{cases} +1 \textit{ with probability } 1/6 \\ 0 \textit{ with probability } 2/3 \\ -1 \textit{ with probability } 1/6 \end{cases} \quad (1)$$

We utilize Achlioptas's proposed equation 1 [1] to generate distribution for creating the random index vector for every term in the structure-constrained content of $CFISC$. For a given document D_i, the index vectors of length l for all the terms corresponding to the given $CFISC_j$ are added. We illustrate this concept of RI on tensor using the Fig. 2, in which we consider a tensor= $\mathbb{R}^{3\times2\times4}$ (in

Fig. 2(a)) with 3 documents, 2 $CFISC$, 4 terms and 7 non-zero entries. The entries in the tensor correspond to the occurrences of a given term in the given $CFISC$ for the document. Using that equation 1, the random index vectors of length 6 for the 4 terms are generated (see in Fig. 2(b). Let us consider document D_1 with three tensor entries $a_{121} = 1$, $a_{123} = 1$ and $a_{124} = 1$ corresponding to $CFISC_1$ and three terms $Term_1$, $Term_3$ and $Term_4$. The random vectors (from Fig. 2(b)) are added to these three terms in D_1. The sparse representation of the resulting vector $(a_{12:})$ for D_1 (given in Fig. 2(c)) contains two non-zero tensor entries $a_{123} = 1$ and $a_{124} = -1$. Fig. 2(d) shows the final reduced tensor $\mathcal{T}_r$ in sparse representation containing 6 non-zero entries.This example demonstrates how our technique can reduce the term space for such a small dataset.

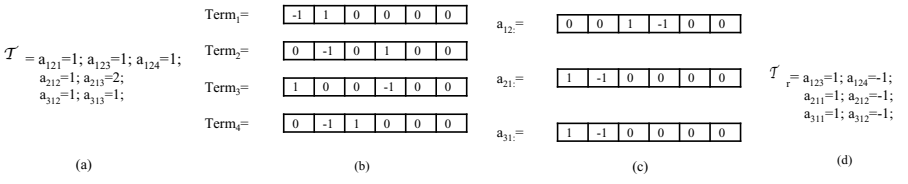


Fig. 2. Illustration of Random Indexing on a 3-order tensor resulting in a randomly reduced tensor $\mathcal{T}_r$

It can be seen that the number of entries in $\mathcal{T}_r$, randomly reduced $\mathcal{T}$, is less in number than its original and maintains the shape of $\mathcal{T}$ as it retains the same similarity between D_2 and D_3. The index vectors in RI are sparse; hence the vectors use less memory store and they are added faster. The randomly-reduced structure-constrained content of $CFISC$ becomes another tensor mode for representing and analyzing XML documents.

3.4 The TSM Representation, Decomposition and Clustering

Given the tensor $\mathcal{T}$, the next task is to find the hidden relationships between the tensor orders. The tensor decomposition algorithms enable an overview of the relationships that can be further used in clustering. However, as already mentioned, most of these decomposition algorithms cannot be applied on very large or dense tensor as the tensors cannot be loaded into memory [13]. To alleviate this problem, the tensors need to be built and unfolded or matricized incrementally. Fig. 3 shows the process of matricization or unfolding along the mode-1 of $\mathcal{T}$ which results in a matrix $\mathrm{T}_{(1)}$. This means that the mode-1 fibers (higher order analogue of rows and columns) are aligned to form a matrix. Essentially this means that the mode-1 fibers of $\mathcal{T}$ are mapped to the rows of matrix $\mathrm{T}_{(1)}$ and the modes-2 and -3 are mapped to the columns of this matrix. We apply the proposed PTCD as shown in Fig. 4 to progressively build and then decompose the tensor using SVD. The motivation for this new tensor decomposition algorithm is that the computations by other decompositions store the fully formed matrices, which are dense and hence cannot scale to very large sized tensors. But

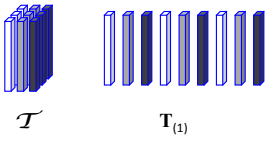


Fig. 3. Mode-1 matricization of a 3-order tensor

Input: Data File: TF, block size: b, number of modes(orders) : M
where $m \in \{1, 2, \ldots, M\}$ and Number of required dimensions: η
Output: *Matricized Tensor*: $\mathbf{T}$
and left singular matrix with η dimensions for 1-mode : $\mathbf{U}_\eta$
1. For every $\mathbf{T}_{(m)} \in \{\mathbf{T}_{(1)}, \mathbf{T}_{(2)}, \ldots, \mathbf{T}_{(M)}\}$
   a. Initialize $\mathbf{T}_{(m)} = \phi$
2. Divide TF into blocks of size b
3. For every block b do
   a. Create tensor $\mathcal{T}_b$
   b. For $m = 1$ to M do
      $\mathbf{T}'_{(m)}$ = Unfold $\mathcal{T}_b$ along its mth mode //Matricize the tensor
      $\mathbf{T}_{(m)} = \mathbf{T}_{(m)} + \mathbf{T}'_{(m)}$ //Update the Mode-m matricized tensor
4. Compute SVD on $\mathbf{T}$ with η dimensions, $\mathbf{T}^\eta = \mathbf{U}_\eta \Sigma_\eta \mathbf{V}_\eta^T$

Fig. 4. Progressive Tensor Creation and Decomposition algorithm (PTCD)

PTCD stores the sparse matrices generated progressively and enables further processing to be performed on the tensor. PTCD builds the tensor progressively by unfolding the tensor entries for the user-defined block size b to a sparse matrix $\mathrm{T}'_{(m)}$ where $m \in \{1, 2, \ldots, M\}$ and M is the number of modes. Then this unfolded matrix, $\mathrm{T}'_{(m)}$ is used to update the final sparse matrix $\mathrm{T}_{(m)}$. After updating, all the tensor entries to the final matrix, $\mathrm{T}_{(m)}$ is then decomposed to the user-defined number of required dimensions η using SVD.

Huang et al. [6] have theoretically proved that HOSVD on a tensor simultaneously reduces the subspace and groups the values in each order. For the 3-order tensor $\mathcal{T}$, the left singular matrix on the document order, $\mathbf{U}_{\eta(D)}$ provides the clustering results on the data index direction; hence they are the cluster indicators for grouping the documents. Consequently, we apply the K-means clustering algorithm on the $\mathbf{U}_{\eta(D)}$ matrix to generate the required number of clusters of the documents.

4 Experiments and Discussion

Experiments are conducted to evaluate the accuracy and scalability performance of XCT on the real-life datasets.

4.1 Datasets

Three real-life XML datasets which have extreme characteristics, INEX 2009 Wikipedia documents collection (Wikipedia)[1], INEX 2006 IEEE [4](IEEE) and

[1] http://www.inex.otago.ac.nz/tracks/wiki-mine/wiki-mine.asp

ACM SIGMOD (ACM)[2,10], were used after a careful analysis of a number of datasets. The INEX 2009 document mining track used the Wikipedia dataset with semantically annotated tags to perform the clustering task. This dataset contains a very large number of documents with deeper structure and a high branching factor. It also supports multi-label categories in which one document can have more than one category. On the other hand, IEEE has single-labeled categories and contains more formatting tags and fewer semantic tags. Finally, the ACM is a small dataset that contains 140 XML documents corresponding to two DTDs, IndexTermsPage.dtd and OrdinaryIssuePage.dtd (with about 70 XML documents for each DTD), similar to the setup in XProj [2]. This dataset has been chosen in order to evaluate our method against other representations and decomposition algorithms which could work only on this kind of small datasets.

4.2 Experimental Design

Experiments were conducted on the High Performance Computing system, with a RedHat Linux operating system, 16GB of RAM and a 3.4GHz 64bit Intel Xeon processor core. Experiments were conducted to evaluate the accuracy of clustering results of XCT over other clustering techniques, decomposition techniques and representation. Previous research for XML documents clustering [2] has used the ACM to cluster the documents into two groups according to their structural similarity. To compare our work with this earlier research, we conducted our experiments not only with two cluster categories according to structural similarity but also on 5 categories using expert knowledge considering both the structure and the content features of XML documents. Due to the small number of terms in this dataset, the random indexing option for XCT has been disabled.

Table 1. Details of the real life datasets

Dataset Names / Attributes	Wikipedia	IEEE	ACM
No. of Docs	54,575	6054	140
No. of tags	34,686	165	38
No. of internal nodes	15,128,407	472,351	2070
Max length of a document	10347	691	45
No. of distinct terms	1,900,072	114,976	7135
Total No. of words	21,480,198	3,695,550	38141
Size of the collection	2.94GB	272MB	1 MB
Presence of formatting tags	Yes	Yes	No
Presence of Schema	Yes	Yes	Yes
Number of Categories	12,803	18	5

Following are the representation and the other existing algorithms used for comparing the outputs of the proposed XCT method.

Structure Only (SO) Representation: An input matrix $D \times CFI$ is generated similar to XProj [2].

Content Only (CO) Representation: The content of XML documents is represented in a matrix $D \times Terms$ with each matrix entry containing term frequency of terms in D.

Structure and Content Representation (S+C) using VSM: The structure and the content features for the documents are represented in a matrix by concatenating the CO and SO representations side by side.

Clustering Using CP and Tucker: The left singular matrix resulting from applying CP or Tucker decomposition on the tensor is used as an input for k-means clustering.

Clustering Using MACH: The MACH decomposition technique has been applied on the original tensor with random indexing. MACH randomly projects the original tensor to a reduced tensor with smaller percentage of entries (10% from the original tensor as specified in [13]) and then uses Tucker decomposition to decompose the reduced tensor. To compare with XCT, we apply k-means clustering on the left singular matrix to group the documents.

Moreover, since the INEX dataset has been used by other researchers: we provide the results cited by other researchers [4,11] as well in our analysis.

4.3 Evaluation Measures

The standard criterion of purity is used to determine the quality of clusters by measuring the extent to which each cluster contains documents primarily from one class. The macro and micro purity of the entire clustering solution is obtained as a weighted sum of the individual cluster purity. In general, the larger the value of purity,the better the clustering solution is.

$$Purity = \frac{\#\ Documents\ with\ the\ majority\ label\ in\ cluster\ k}{\#\ Documents\ in\ cluster\ k} \tag{2}$$

$$Micro - Purity = \frac{\sum_{k=0}^{n} Purity(k) * \#\ Documents\ Found\ By\ Class(k)}{\sum_{k=0}^{n} \#\ Documents\ Found\ By\ Class(k)} \tag{3}$$

$$Macro - Purity = \frac{\sum_{k=0}^{n} Purity(k))}{Total\ Number\ of\ Categories} \tag{4}$$

4.4 Empirical Analysis

Accuracy of Clustering: Tables 2 and 3 provide the purity results of clustering on the datasets using XCT, other representations and other decomposition algorithms. As can be seen from these three tables, the proposed XCT method not only outperforms our benchmarks but also other INEX submissions in terms of the accuracy of their clustering solution. It should be noted that algorithms

Table 2. Clustering results on Wikipedia and IEEE

Methods	#50 clusters		#100 clusters	
	Micro-purity	Macro-purity	Micro-purity	Macro-purity
XCT	0.13	0.14	0.14	0.13
S+C using VSM	0.13	0.14	0.13	0.14
Clustering using MACH	0.087	0.089	0.089	0.088
CO	0.10	0.12	0.12	0.13
SO	0.09	0.11	0.11	0.10
BilWeb-CO [11]	NA	NA	0.10	0.13

Methods	Micro-purity	Macro-purity
XCT	0.23	0.23
S+C using VSM	0.18	0.14
Clustering using MACH	0.17	0.20
CO	0.10	0.12
SO	0.08	0.10
Nayak et. al [4]	NA	0.18
Doucet et. al [4]	NA	0.13

Table 3. Results of clustering on ACM

Methods	#2 clusters		#5 clusters	
	Micro-purity	Macro-purity	Micro-purity	Macro-purity
XCT	1	1	0.91	0.91
S+C using VSM	0.98	0.98	0.75	0.79
Clustering using MACH	0.97	0.93	0.70	0.75
Clustering using Tucker	0.56	0.48	0.59	0.87
Clustering using CP	0.89	0.93	0.84	0.93
CO	0.97	0.94	0.73	0.78
SO	1	1	0.64	0.72

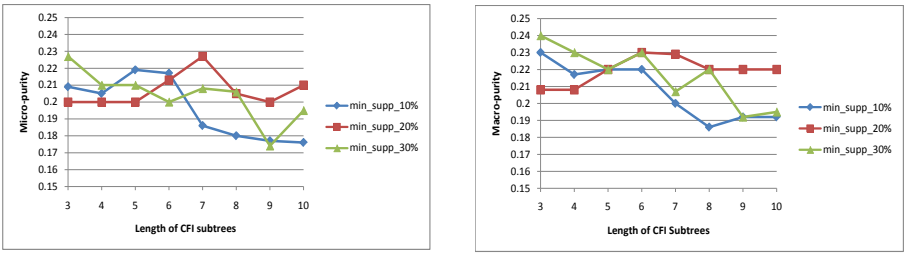


Fig. 5. Sensitivity of length constraint on the micro-purity values for IEEE

such as CP, Tucker could not scale even to the medium-sized dataset, IEEE and hence their results were not reported but the proposed PTCD was able to decompose even large dataset as shown in Tables 2 and 3. As the categories in IEEE was based on both the structure and the content we utilized this dataset for analyzing the sensitivity of the length constraint and *min_supp* values on the purity. We conducted experiments by varying the length constraint(***len***) of the CFI subtrees from 3 to 10 for support thresholds from 10% to 30%. From Fig. 5 which indicates that with the increase in the length constraint the micro-purity and macro-purity values drops especially at 10% and 30% support threshold. Also, length constraint of over 7 shows a negative impact on the purity. With longer length patterns the content corresponding to the CFI subtrees becomes specific and hence results in less accuracy than the content corresponding to shorter subtrees. This shows the suitability of constraining the CFI subtrees as in PCITMinerConst.

Time Complexity Analysis: The time complexity of XCT is composed of five major components, namely frequent mining for CFI subtrees, clustering of CFI subtrees, random indexing, matricization and decomposition in PTCD. It is given by $O(dsm) + O(drp) + O(tk\gamma) + O(dr\gamma) + O(dk'\gamma)$ where d represents the number of documents, s is the number of 1-Length CFI subtrees, m is the number of PCITMinerConst iterations, r is the number of structure-based clusters, p is the number of similarity computation iterations, γ is the size of the random index vector, k and k' are the non-zero entries per column in the tensor before random indexing and in the matricized tensor after random indexing respectively. The time complexity of PTCD is $O(dr\gamma) + O(dk'\gamma)$, which includes the cost of matricization along the *mode-n* and the sparse SVD.

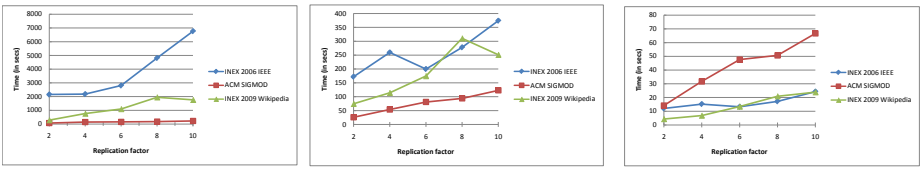


Fig. 6. Scalability of PTCD

Scalability Test: All the three datasets were used for this analysis with *min_supp* at 10%, ***len*** at 5, γ at 100 and the number of clusters chosen, 5, 18 and 100 respectively. Also we used 1000 documents each for IEEE and Wikipedia. The execution time of PCITMinerConst is less than a few 10 milliseconds and hence it has not been reported as it is not of much significance. We can see from Fig. 6 that both XCT and PTCD scale nearly linearly with the dataset size. The PTCD algorithm includes two main steps: (1)loading the tensor file into memory by matricization, and (2) decomposing the matrices using SVD. As it can be seen from Fig. 6 that minimal time is spent on decomposition and a large chunk of time on loading the tensor file into memory. Also, it is interesting to note the PTCD for ACM is greater in comparison to IEEE which indicates that random indexing option in XCT method helps to reduce the complexity of decomposition by reducing the term space. An interesting problem is how to choose the value γ for the seed length in random indexing. The Johnson- Lindenstrauss result was used to get the bounds for γ as given by $\gamma = \lceil 4(\epsilon^2/2 - \epsilon^3/3)^{-1} ln\ n \rfloor$. However, we found that for Wikipedia dataset with ϵ=0.5, γ is 433 but in the experiments, $\gamma \approx 100$ was sufficient to obtain good accuracy similar to the results by [3].

5 Conclusion

In this paper, we have proposed a clustering method, XCT, for effectively combining both the structure and the content features in XML documents using TSM model. The experimental results clearly ascertain that XCT outperforms other existing approaches in improving accuracy. Also, our proposed decomposition algorithm PTCD has demonstrated that it has potential for decomposing tensors effectively and could scale for very large datasets. Our future work will focus on reducing the complexity of XCT and applying it on various types of other types of semi-structured documents.

References

1. Achlioptas, D., Mcsherry, F.: Fast computation of low-rank matrix approximations. J. ACM 54(2), 9 (2007)
2. Aggarwal, C.C., Ta, N., Wang, J., Feng, J., Zaki, M.: Xproj: a framework for projected structural clustering of xml documents. In: KDD, pp. 46–55. ACM, USA (2007)

3. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: KDD, pp. 245–250. ACM, New York (2001)
4. Denoyer, L., Gallinari, P.: Report on the xml mining track at inex 2005 and inex 2006: categorization and clustering of xml documents. SIGIR Forum 41(1), 79–90 (2007)
5. Acar, E., Yener, B.: Unsupervised multiway data analysis: A literature survey. IEEE TKDE 21(1), 6–20 (2009)
6. Huang, H., Ding, C., Luo, D., Li, T.: Simultaneous tensor subspace selection and clustering: the equivalence of high order svd and k-means clustering. In: KDD, pp. 327–335. ACM, New York (2008)
7. Jegelka, S., Sra, S., Banerjee, A.: Approximation algorithms for tensor clustering. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 368–383. Springer, Heidelberg (2009)
8. Jimeng, S., Dacheng, T., Spiros, P., Philip, S.Y., Christos, F.: Incremental tensor analysis: Theory and applications. ACM TKDD 2(3), 1–37 (2008)
9. Kolda, T.G., Sun, J.: Scalable tensor decompositions for multi-aspect data mining. In: ICDM, pp. 363–372 (2008)
10. Kutty, S., Nayak, R., Li, Y.: HCX: an efficient hybrid clustering approach for XML documents. In: DocEng, pp. 94–97. ACM, USA (2009)
11. Nayak, R., de Vries, C., Kutty, S., Geva, S.: Overview of the INEX 2009 XML mining track: Clustering and classification of XML documents. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 366–378. Springer, Heidelberg (2010)
12. Sahlgren, M.: An introduction to random indexing. In: Methods and Applications of Semantic Indexing Workshop, TKE (2005)
13. Tsourakakis, C.E.: MACH: Fast Randomized Tensor Decompositions. In: SDM, USA, pp. 689–700 (2010)
14. Vercoustre, A.-M., Fegas, M., Gul, S., Lechevallier, Y.: A flexible structured-based representation for XML document mining. In: Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.) INEX 2005. LNCS, vol. 3977, pp. 443–457. Springer, Heidelberg (2006)

An Efficient Pre-processing Method to Identify Logical Components from PDF Documents

Ying Liu[1], Kun Bai[2], and Liangcai Gao[3]

[1] Department of Knowledge Service Engineering, KAIST
yingliu@kaist.edu
[2] IBM Research T.J. Watson Research Center
kunbai@us.ibm.com
[3] Institute of Computer Science and Technology, Peking University
gaoliangcai@icst.pku.edu.cn

Abstract. As the rapid growth of the scientific documents in digital libraries, the search demands for the documents as well as specific components increase dramatically. Accurately detecting the component boundary is of vital importance to all the further information extraction and applications. However, document component boundary detection (especially the table, figure, and equation) is a challenging problem because there is no standardized formats and layouts across diverse documents.

This paper presents an efficient document preprocessing technique to improve the document component boundary detection performance by taking advantage of the nature of document lines. Our method easily simplifies the component boundary detection problem into the sparse line analysis problem with much less noise. We define eight document line label types and apply machine learning techniques as well as the heuristic rule-based method on identifying multiple document components. Combining with different heuristic rules, we extract the multiple components in a batch way by filtering out massive noises as early as possible. Our method focus on an important un-tagged document format – PDF documents. The experimental results prove the effectiveness of the sparse line analysis.

Keywords: Preprocessing, Boundary Detection, Sparse Line Property, PDF documents, Table and Equation.

1 Introduction

Most prior works on information extraction in digital documents have focused on extracting information from texts. However, often, the most important information being reported in a digital document is presented in multiple components (e.g., tables, equations, figures, algorithms, etc.) and special sections (e.g., abstract and conclusion). These components are widely used in web pages, scientific documents, financial reports, etc. Researchers always adopt these components to introduce a new theory or an important algorithm, to display the latest experimental results, the valuable statistical information, or a summary

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 500–511, 2011.

of their exciting findings in a condensed fashion. Other researchers, for example, who are conducting an empirical study in the same topic, can quickly obtain valuable insights via examining these components.

Although these components are ubiquitous with a history that pre-dates that of sentential texts, they are far from been fully analyzed and utilized, especially the tables and equations. Along with the explosive development of the digital library and internet, these components have became a valuable information source for information seeking and data analysis. Based on the increasing demands to unlock the information inside, more applications appear, e.g., the table search [12] or equation understanding. Considering the life cycles of these components, they are generated and sometimes modified to fit new regulations or to display in new devices. With the content repurposing, each component can be viewed and shared by more people. If the data reported in these components can be extracted and stored in a database, the data can be queried and integrated with other data using database management systems. In order to prepare the data source for further applications, accurately detecting the boundaries of these components plays a crucial role for the later information extraction. While there has been some research on component boundary detection, most them only focus on a single component each time. Although approaches on component analysis are diverse, they share two analyzing steps: boundary detection and structure decomposition. For the further content storage and sharing (e.g., the table data extraction or the equation search), locating the component boundary is the first and crucial step.

In this paper, we propose a novel, efficient and universal method to detect the boundaries of all the listed specific components. Comparing with the previous works, our method has the following contributions:

1) Although document component boundary detection is not a new topic, most previous works only focus on an individual component and it is difficult to apply their methods/algorithms to another different component without much modification. Our method can detect the boundary of a batch of important but different components from a document in one simple iteration. In this paper we focus on the table and equation boundary detection. However, other components e.g., figures and algorithms can also be detected simultaneously.

2) In general, the component boundary detection problem can be transformed into the problem of identifying the lines of the target components, which constitute the component boundaries. With observation of multiple components with diverse layouts from different documents, we notice an interesting phenomenon according to the nature of document layouts, almost all the lines within the component boundaries share an important property: *majority lines belonging to the specific component components are sparse in terms of the text density.* Based on this interesting property, most document logical components can be identified together with different features. Existing *filter-out* based component discovering methods identify the target lines from the entire set of document lines according to certain rules, which in turn results in low *recall*. However, for applications such as table search [12], *recall* is more important than *precision* because once the true

target lines are removed, it is difficult to retrieve them back. Fortunately, the false positive rate can be easily lowered in later structure decomposition step. In this paper, we propose a novel but effective method to quickly locate the component boundary by taking advantage of the aforementioned property. We also propose an *exclusive* based method for identifying component lines, which generates high recall and saves substantial effort to analyze the noisy lines.

3)The PDF format is commonly used for the exchange of documents on the Web and there is a growing need to understand and extract or repurpose data held in PDF documents. Many systems for processing PDF files use algorithms designed for scanned documents, which analyze a page based on its bitmap representation. We believe this approach to be inefficient. Not only does the rasterization step cost processing time, but information is also lost and errors can be introduced. Although we can convert PDF documents into other media (e.g., html, text), then analyze the new documents using the existing methods, identifying various component logical components as well as the contents [2] is still a challenging problem. The major difficulties come from the following aspects: most PDF documents are untagged and do not have basic high-level document logical structural information, which makes the reuse or modification of the documents difficult. In addition, almost all the PDF text extraction tools have the text sequence errors. If converting PDF documents into other media (e.g., image), new noises can be generated by some necessary tools (e.g., OCR).

The rest of the paper is organized as follows. Section 2 reviews several relevant studies in component boundary detection area and the applied machine learning methods in this field. Section 3 introduces the sparse-line property we proposed, describes in detail the sparse line detection and the noise line removing using the conditional random field and support vector machine (SVM) techniques. We also elaborate the label types and the feature sets. Section 4 explains how to locate the component boundary based on the labeled lines as well as other important heuristic rules, e.g., keywords. The detailed experimental results are displayed in Section 5. We conclude our paper with plans for future work in Section 6.

2 Related Works

Chao et al. [2] reported their work on extracting the layout and content from PDF documents. Hadjar et al. developed a tool for extracting the structures from PDF documents. They believe that, to discover the logical components of a document, all/most of the page objects listed by PDF document content stream need to be analyzed (e.g., text objects, image objects, path objects). However, the object overlapping problem happens frequently. If all the objects are analyzed, more effort needs to be spent to firstly segment them from each other. In addition, even we identified these objects, they are still too high level to fulfill many special goals, e.g., detecting the tables, figures, mathematical formulas, footnotes, references, etc. Instead of converting the PDF documents into other media types (e.g., image or HTML) and then applying the existing techniques, we process PDF documents directly from the text level.

In the past two decades, a good number of researches have been done to discover the document layout by converting the PDFs to image files. However, the image analysis step can introduce noise (e.g., some text may not be recognized or some images may not be correctly recognized). In addition,because of the limited information in the bitmap images, most of them only work on some specific document types with minimal object overlap: e.g., business letters, technical journals, and newspapers. Some researchers combine the traditional layout analysis on images with low-level content extracted from the PDF file. Even if the version 6 of PDF allows a user to create a file containing structure information, most of them do not contain such information. In this paper, we propose a method that relies solely on the PDF extracted content, not longer requiring the conversion to any other document medium and apply any further processing methods.

3 The Sparse-Line Property

Although the PDF was designed as a powerful fixed-layout format that allows documents created within any desktop publishing package to be viewed in the original typeset design, regardless of the systems where it is being displayed, there were trends to recover the document structural layouts and to identify logical components from PDFs. To successfully get the data from an important component, detecting the component boundary is a crucial step. Based on the observation, we notice that different lines in a same document page have different widths, text densities, and the sizes of the internal spaces between words. A document page contains at least one column. Many journals/conferences require two (e.g., ACM and IEEE templates) or three even four columns. Some document lines have the same length as the width of the regular document column (e.g., most lines in this paper), some others are much longer (e.g., cross over multiple document columns) or shorter. Based on the size of the internal spaces within a document line, the majority of document lines contain normal space sizes between two adjacent words while some lines have large spaces.

Sparse line is originally proposed in [14]. Because we extend the table component to all the document logical components in this paper, we modify the *sparse line* definition as follow. A document line is a sparse line if either condition is satisfied: 1) The maximum space gap between a pair of consecutive words within the line is larger than a threshold *sg*; 2) The length of the line is shorter than a threshold *ll*. Different " *ll*" may generate different sparse line labeling results. We define it as the half of the regular document column width.

Figure 1 shows a snapshot of a PDF document page as an example. We highlight the sparse lines in red rectangles and the lines with specific keywords in blue rectangles. Apparently, the table/ equation/ figure body-content lines are labeled as sparse lines according to the definition. Four sparse lines are not located within the component boundary we pre-defined: one caption lines and three short lines that are the last line in a paragraph. We label them as sparse lines because they satisfy the second condition. Heading/footer lines may also

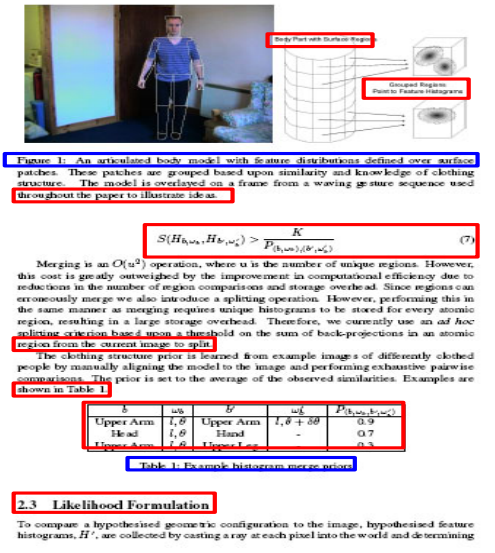


Fig. 1. An example PDF page

belong to this category. Since such short-length lines also happen in some table rows with only one filled cell, we consider them as sparse lines to avoid missing out the potential table lines. Such noise sparse lines are very few because they usually only exist at the headings or the last line of a paragraph. In addition, the short length restriction also reduce the frequency. We can easily get rid of them based on the coordinate information later. Comparing with [14], which automatically identified the table boundary based on sparse lines analysis, this paper aims to identify multiple document components (tables, equations, figures, etc.) simultaneously.

3.1 Machine Learning Methods

In this paper, we apply two machine learning methods – Support Vector Machine (SVM) [1] and Conditional random fields (CRF) [11] on the component boundary detection Furthermore, we elaborate the feature selection, analyze the factor effects of different features, and compare the performance of CRF/SVM approaches with our proposed rule-based method. Different from most CRF applications, the unit of our problem is a document line, instead of a single word. Before classifying the document lines, we have to construct the lines first. Our bottom-up line construction approach can be found in [14].

Overall, our features can be classified into three categories: the *orthographic features*, the *lexical* features, and the *document layout features*. Most related works treat the vocabulary as the simplest and most obvious feature set. Such features, named as orthographic features, define how these input data appear based on regular expressions as well as prefixes and suffixes. For the table boundary detection, layout features are the most important parts. However, for this

paper, we found that the first two types are more crucial because of the nature of document components. The detailed features can be found in [14].

Different from the traditional component boundary detection works, we use an exclusive method to label all the potential target lines. Each document line will be examined and labeled with a predefined category. For the boundary detection of a specific document component, we can easily identify the noisy document zones and remove them as early as possible.

4 Component Boundary Detection

After labeling each line using machine learning methods, Figure 2 shows the distribution of the sparse lines in a document page. In order to detect multiple component boundaries in a batch way, we adopt a universal boundary detection method, which contains the follow four steps:

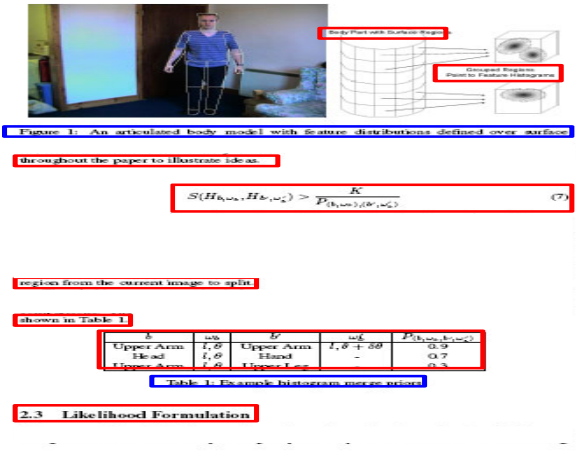


Fig. 2. The sparse lines in a PDF page

1) Locating one end of the boundary by detecting the pre-defined keywords. Although caption lines are not included in the boundary, the keywords are usually powerful indicators, which facilitates the boundary detection.

2) Deciding the other end of the boundary by judging the caption location. Although most caption lines are not included in the boundary area, the caption location (below/top) is crucial to find the other side of the boundary, especially when multiple components appear in a same page altogether.

3) Analyzing the collected sparse lines between two ends; We focus on different aspects for different components. For example, tabular structure is the key feature of a table. Because of the inherited sequence error problem of PDF text extraction tools, the line sequence resorting work usually happens in this step.

4) Removing noisy sparse lines and retrieving the positive lines labeled as non-sparse lines that are filtered out in the early stage, if needed. Usually the

noisy sparse lines are located just beyond the component boundary. Using table boundary as an example, a typical positive line to be retrieved is the long table lines (e.g., long nesting table lines).

Because of the limited space, we use table and equation boundary detection as two examples. We define the main table content rows as the table boundary, which does not include the table caption and the footnote. In order to facilitate the table starting boundary detection, we define a keyword list, which contains all the possible starting keywords of table captions, such as "Table, TABLE, Form, FORM," etc. Most tables have one of these keywords in the captions. If more than one tables are displayed together, the keyword is very useful to separate the tables from one another. Once we detect a line (not only the sparse line) starting with such a keyword, we treat it as a table caption candidate. Then we check all the nearby sparse lines and merge them with the sparse area according to the vertical distances between adjacent lines. Tabular structure within the sparse area is the key feature to verify a table boundary.

Comparing with other components, the equation boundary always contains a set of sparse lines which satisfy some particular patterns: short lengths, disordered sequences, inconsistent font information, and single characters or words together with mathematic operation symbols. In addition, the most important difference between equation boundary and table boundary is that there is no obvious tabular structure within an equation.

In previous researches with imbalanced data, *recall* is usually lower than *precision* for the true class. However, in the document component boundary detection problem, *recall* is more important than *precision* at least in the information retrieval (IR) field, because collecting the real target lines as more as possible is crucial to the overall boundary detection accuracy. The texts within the detected component boundary will be analyzed carefully in the later structure decomposition phase. Once we locate a component boundary, we zoom in the detected boundary and try to retrieve the missing lines (e.g, long table lines) that are labeled as non-sparse lines or to remove the false lines (e.g., surrounding noisy sparse lines) to improve the *recall*.

5 Experiments and Results

In this section, we demonstrate the experimental results of evaluating the document component boundary detection with two machine learning methods Our experiments can be divided into four parts: the performance evaluation of different methods, different feature settings, different datasets, and different parameter settings.

5.1 Data Set

Instead of analyzing document components from a specific domain, we aim to collect components as much different varieties as possible from digital libraries. We continue our experiments based on the same data sources in [14]: chemical

scientific digital libraries (Royal Chemistry Society[1]), Citeseer[2], and archeology[3] in chemistry, computer science and archeology fields. The size of each PDF repository we collected exceeds 100,000, 10,000 and 8,000 respectively in terms of scientific papers. We extended the size of our training set to 450 randomly selected pages. Among these pages, we refer 150 pages from the chemistry field as the dataset H, 150 pages from the computer science field as the dataset S, and 150 come from the archeology field as the dataset A. The total number of the lines in three datasets are 17035, 20468, and 13945 respectively. For every document line, we manually identify it with a defined label. In order to get an accurate and robust evaluation on the component boundary detection performance, we adopt a hold-out method by randomly dividing the dataset into five parts and in each round we train four of the five parts and tested on the remaining one part. The final overall performance comes from the combined five results. In our experiment, we use the java-implemented, first-order CRF implementation – Mallet – to train two versions of the CRF with binary features and the actual values. For SVM, we adopt SVM light [7].

Because this work can be viewed as an extension of our previous work in [14], the results of the line construction can be found there. In this paper, we mainly test the performance of line labeling for each specific component by considering the pre-defined keywords and rules.

5.2 Performance of Sparse Line Detection

We perform a five-user study to evaluate the quality of the sparse line detection. Each user checks the detected sparse lines in 30 randomly selected PDF document pages in each dataset. The evaluation metrics are *precision* and *recall*, which are defined in [14]. The results are listed in Table 2.

Table 1. The performance evaluation of the sparse line detection

datasets	H	A	S
The Number of PDF pages	150	150	150
Recall of sparse line detection	99.75	99.40	99.38
Precision of sparse line detection	98.33	98.75	98.68

There are two reasons for the potential errors: 1) some tables have long cells or very small spaces between the adjacent table columns because of the crowd layout. 2) The mathematical equation boundary may include some surrounding noisy sparse lines. In order to include the correct lines in each component, we regulate thresholds by setting ll with a tolerate value and sp with a smaller value. The trade off is mislabeling some non-sparse lines as sparse lines. However, we have further steps to do the boundary structure analysis, including such non-sparse or noisy lines(low *precision*) is not a big problem. Within the datasets H,

[1] http://www.rsc.org/
[2] http://citeseer.ist.psu.edu/
[3] http://www.saa.org/publications/AmAntiq/AmAntiq.html

A and S, 80.25% lines are labeled as non-sparse lines and can be easily removed as noise.

5.3 Performance of Noise Line Removal

Different from [14], the noise lines here refer all the sparse lines that are not belong to the target component boundary, e.g., the last sparse line in Figure 2. The input of this step is all the detected sparse lines while the output should be the filtered sparse lines that are located in the components. Because of the limited space, we use equation and table as examples.

Table 2 lists the noise removal results, measured in *precision* and *recall*. FP refers the beginning dataset – all lines in a page. RNS refers the non-sparse lines. RH refers the noisy heading lines. HF refers the noisy header and footnote lines. CAP is the noisy caption lines, REF is the noisy reference lines, and PP represents the postprocessing step. Along with the noise removing, the size of the sparse line dataset decreases and the *precision* of the table/equation line labeling increase steadily. Non-sparse line removing and the postprocessing are two crucial steps for the table/equation boundary detection problem. The results on three datasets are consistent without any remarkable difference. The *precision* value is improved from 10.48% to 61.47% after removing all noises. In addition, the further steps are much easier because of the dramatically reduced sparse line set. Although the results are still not satisfying, the remaining false positive table/equation lines scatter in the page and the large distance to the table caption is an important feature to identify the table lines.

Table 2. The performance evaluation of the noisy sparse line removal

D/NLC	H precision	H recall	A precision	A recall	S precision	S recall
FP	10.79	100.00	11.25	100.00	10.48	100.00
RNS	43.18	99.20	43.76	99.25	42.55	98.21
RH	53.60	99.16	55.84	99.06	53.13	98.08
HF	55.76	99.13	56.98	98.92	55.32	97.80
CAP	57.55	99.06	58.10	98.55	57.28	97.25
REF	60.83	98.97	61.47	98.83	60.79	97.36
PP	98.33	98.85	98.75	98.80	98.68	97.28

Table 2 also shows the *recall* curves with the same experimental conditions. The initial *recall* values are 100% because no line is removed. Along with each step, the recall is decreasing because few true table/equation lines are mislabeled and removed. Within three datasets, dataset S has the worst recall value because most computer science documents do not follow the standard template strictly and some true table lines are mislabeled.

5.4 Table/Equation Boundary Detection

Although the remaining sparse line set contains almost all the table lines and the equation lines, the accuracy is still not satisfying. the typical false positive table lines are the lines with short length. Such lines are usually located at the end

of paragraphs, the last line of a table caption, or a short table footnote without special beginning symbol etc. Considering the distance features, most of the first type can be filtered out. For those missed true table lines, analyzing the location information of adjacent sparse line sections together with the table caption help us to retrieve them back. With these heuristical rules, the *precision* values is enhanced to 95.79% and the *recall* values is close to 98.61%. After we decide the table boundary, we treat all the remained of the sparse lines as the potential equation boundary. Based on the dataset in section 5.1, the *recall* of equation boundary detection is 95.25 and the *precision* is 87.85. Because we will launch the equation structure decomposition (see details in another paper) based on the detected equation boundary, we welcome high *recall* and the relatively low *precision* will be improved easily.

6 Conclusions

In this paper, we propose a preprocessing method to detect the boundaries of a batch of different document components. With an interesting observation of the document lines, we simplify the document boundary detection problem into a line labeling problem by considering the nature of document lines. The whole method can be viewed as a classification and filtering sequence. As the noisy lines are removed, we can easily detect the target lines to constitute the components. Combining different heuristic rules, our method is applicable to detect multiple document components.

References

1. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Journal Data Mining and Knowledge Discovery 2(2), 121–167 (1998)
2. Chao, H., Fan, J.: Layout and content extraction for PDF documents. In: Marinai, S., Dengel, A.R. (eds.) DAS 2004. LNCS, vol. 3163, pp. 213–224. Springer, Heidelberg (2004)
3. Chen, S.T.H., Tsai, J.: Mining tables from large scale html texts. In: Proc. 18th International Conference Computational Liguistics, Saarbrucken, Germany (2000)
4. Ha, J., Haralick, R., Philips, I.: Recursive x-y cut using bounding boxes of connected components. In: Proc. Third International Conference Document Analysis and Recognition, pp. 952–955 (1955)
5. Hurst, M.: Layout and language: Challenges for table understanding on the web. In: Proceedings of the International Workshop on Web Document Analysis, pp. 27–30 (2001)
6. Shin, N.G.J.: Table recognition and evaluation. In: Proceeding of the Class of 2005 Senior Conference, Computer Science Department, Swarthmore College, pp. 8–13 (2005)
7. Joachims, T.: Svm light, `http://svmlight.joachims.org/`
8. Kieninger, T., Dengel, A.: Applying the t-rec table recognition system to the business letter domain. In: In Proc. of the 6th International Conference on Document Analysis and Recognition, pp. 518–522 (September 2001)

9. Kieninger, T.G.: Table structure recognition based on robust block segmentation. In: Proceeding of Document Recognition V, SPIE, vol. 3305, pp. 22–32 (January 1998)
10. Krupl, B., Herzog, M., Gatterbauer, W.: Using visual cues for extraction of tabular data from arbitrary html documents. In: Proceeding of the 14th International Conference on World Wide Web, pp. 1000–1001 (2005)
11. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. 18th ICML, pp. 282–289. Morgan Kaufmann, San Francisco (2001)
12. Liu, Y., Bai, K., Mitra, P., Giles, C.L.: TableSeer: Automatic Table Metadata Extraction and Searching in Digital Libraries. In: ACM/IEEE Joint Conference on Digital Libraries, JCDL, pp. 91–100 (2007)
13. Liu, Y., Bai, K., Mitra, P., Giles, C.L.: Improving the table boundary detection in pdfs by fixing the sequence error of the sparse lines. In: 10th International Conference on Document Analysis and Recognition, ICDAR 2009 (2009)
14. Liu, Y., Mitra, P., Giles, C.L.: Identifying Table Boundaries in Digital Documents via Sparse Line Detection. In: CIKM 2008, Napa Valley, California (2008)
15. McCallum, A.: Efficiently inducing features of conditional random fields. In: Nineteenth Conference on UAI (2003)
16. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields. In: CONLL 2003 Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, vol. 4 (2003)
17. Ng, H., Lim, C., Koo, J.: Learning to recognize tables in free text. In: ACL Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics (1999)
18. Ng, H., Lim, C.Y., Koo, J.T.: Learning to recognize tables in free text. In: Proc. of the 37th Annual Meeting of the Association of Computational Linguistics on Computational Linguistics, pp. 443–450 (1999)
19. Penn, G., Hu, J., Luo, H., McDonald, R.: Flexible web document analysis for delivery to narrow-bandwidth devices. In: Sixth International Conference on Document Analysis and Recognition (2001)
20. Pinto, D., McCallum, A., Wei, X., Bruce, W.: Table extraction using conditional random fields. In: Proceeding of Proceedings of the 26th ACM SIGIR, Toronto, Canada (July 2003)
21. Safavian, S., Landgrebe, D.: A survey of decision tree classifier methodology. SMC 21(3), 660–674 (1991)
22. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: NAACL 2003, Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, vol. 1 (2003)
23. Shamilian, J., Baird, H., Wood, T.: A retargetable table reader. In: Proc. of the 4th Int'l Conf. on Document Analysis and Recognition, pp. 158–163 (1997)
24. Stoffel, A., Spretke, D., Kinnemann, H., Keim, D.A.: Enhancing document structure analysis using visual analytics. In: SAC 2010 Proceedings of the 2010 ACM Symposium on Applied Computing (2010)
25. Wang, J., Hu, J.: A machine learning based approach for table detection on the web. In: The Eleventh International World Wide Web Conference 2002, pp. 242–250 (November 2002)
26. Wang, Y., Hu, J.: Detecting tables in html documents. In: Proc. of the 5th IAPR DAS, Princeton, NJ (2002)

27. Wang, Y., Philips, I., Haralick, R.: Automatic table ground truth generation and a background-analysis-based table structure extraction method. In: Proc. of the 6th Int'l Conference on Document Analysis and Recognition, p. 528 (September 2001)
28. Yildiz, B., Kaiser, K., Miksch, S.: pdf2table: A method to extract table information from pdf files. In: Proceedings of the 2nd Indian International Conference on Artificial Intelligence IICAI 2005, Pune, India (2005)
29. Yoshida, M., Torisawa, K., Tsujii, J.: A method to integrate tables of the world wide web. In: Proceedings of the International Workshop on Web Document Analysis (WDA 2001) (2001)
30. Zanibbi, R., Blostein, D., Cordy, J.: A survey of table recognition: Models, observations, transformations, and inferences. Int'l J. Document Analysis and Recognition 7(1), 1–16 (2004)
31. Zheng, Z.: Naive bayesian classifier committees. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 196–207. Springer, Heidelberg (1998)

Combining Proper Name-Coreference with Conditional Random Fields for Semi-supervised Named Entity Recognition in Vietnamese Text

Rathany Chan Sam, Huong Thanh Le,
Thuy Thanh Nguyen, and Thien Huu Nguyen

Hanoi University of Science and Technology,
1 DaiCoViet street, Hanoi, Vietnam
rathany_cam@yahoo.com,
{huonglt,thuynt}@it-hut.edu.vn,
nguyenhuuthien88bk@yahoo.com

Abstract. Named entity recognition (NER) is the process of seeking to locate atomic elements in text into predefined categories such as the names of persons, organizations and locations. Most existing NER systems are based on supervised learning. This method often requires a large amount of labelled training data, which is very time-consuming to build. To solve this problem, we introduce a semi-supervised learning method for recognizing named entities in Vietnamese text by combining proper name coreference, named-ambiguity heuristics with a powerful sequential learning model, Conditional Random Fields. Our approach inherits the idea of Liao and Veeramachaneni [6] and expands it by using proper name coreference. Starting by training the model using a small data set that is annotated manually, the learning model extracts high confident named entities and finds low confident ones by using proper name coreference rules. The low confident named entities are put in the training set to learn new context features. The F-scores of the system for extracting "Person", "Location" and "Organization" entities are 83.36%, 69.53% and 65.71% when applying heuristics proposed by Liao and Veeramachaneni. Those values when using our proposed heuristics are 93.13%, 88.15% and 79.35%, respectively. It shows that our method is good in increasing the system accuracy.

Keywords: information extraction, named entity extraction, entity coreference, semi-supervised learning, CRFs.

1 Introduction

Named Entity Recognition is a subtask of information extraction. Its purpose is to identify and classify certain proper nouns into some predefined target entity classes such as person, organization, location and temporal expressions.

Much previous work on NER followed the supervised learning approach [2], [3], [9], [12], [15] which requires a large hand-annotated corpus. Such approaches

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 512–524, 2011.

can achieve good performances. However, annotating such a corpus requires a lot of human effort. This problem can be solved by using a sequence-based semi-supervised method that trains a classification model on an initial set of labelled data, makes predictions on a separate set of unlabelled data, and then iteratively attempts to create an improved model using predictions of the previously generated model (plus the original labelled data). Based on this method, we propose a semi-supervised learning method for recognizing named entities in Vietnamese text by combining proper name coreference, named-ambiguity heuristics with a powerful sequential learning model, Conditional Random Fields (CRFs). Our approach inherits the idea of Liao and Veeramachaneni [6] and expands it by using proper name coreference. Starting by training the model using a small data set that is tagged manually, the learning model extracts high confident named entities with and finds low confident NEs by using proper name coreference rules. The low confident NEs are put in the training data set to learn new context features.

Example 1.

(a) *Hôm nay có mưa lớn ở* ***Thành phố Hồ Chí Minh*** /It rains heavily in Hochiminh city today.

(b) *Chính phủ đang tìm giải pháp chống tắc đường ở* ***TP HCM*** /The government is finding a method to solve traffic jam in Hochiminh city.

In Example 1, both "*Thành phố Hồ Chí Minh*/Hochiminh city" and "*TP HCM*" are Location entities and refer to one location. However, the system can only find one Location entity with high confident score, which is "*Thành phố Hồ Chí Minh*/Hochiminh city". The phrase "*TP HCM*" is not recognized as a Location entity by the system since the confidence score of this phrase is smaller than the threshold. Based on the coreferent rules, the system discovers that "*Thành phố Hồ Chí Minh*/Hochiminh city" and "*TP HCM*" refer to the same location. From that point of view, "*TP HCM*" is considered as a low confidence part of "*Thành phố Hồ Chí Minh*/Hochiminh city". "*TP HCM*" is then forced to be a Location entity. It is put in the training set to relearn the new feature context.

In addition, based on our empirical study, several named entity (NE) rules are manually added to the system, in order to create new training data from unlabelled text.

The rest of this paper is organized as follows. Section 2 introduces recent studies on semi-supervised NER methods and works that inspire our research. Section 3 briefly introduces include CRF and the training and inference of the CRF. Section 4 discusses the semi-supervised NER problem for Vietnamese text and our solution to this problem. Section 5 analyzes our experimental results. Finally, our conclusions and future work are given in Section 6.

2 Related Works

The term "semi-supervised" (or "weakly supervised") is relatively recent. The main technique for semi-supervised learning is called "bootstrapping" and involves a small degree of supervision for starting the learning process.

Niu et al [13] present a bootstrapping approach for NER. This approach only requires a few common noun/pronoun seeds that correspond to the concept for the target NE type, e.g. *he/she/man/woman* for Person entity. The entire bootstrapping procedure is implemented as training two successive learners: (i) a decision list is used to learn the parsing-based high precision NE rules; (ii) a Hidden Markov Model is then trained to learn string sequence-based NE patterns. The second learner uses the training corpus automatically tagged by the first learner.

Mohit and Hwa [11] used Expectation Maximization (EM) algorithm along with their Naïve Bayes classifier to form a semi supervised learning framework. In this framework, the small labelled dataset is used to do the initial assignments of the parameters for the Naïve Bayes classifier. After this initialization step, in each iteration the Naïve Bayes classifier classifies all of the unlabelled examples and updates its parameters based on the class probability of the unlabelled and labelled NE instances. This iterative procedure continues until the parameters reach a stable point. Subsequently, the updated Naïve Bayes classifies the test instances for evaluation.

Perrow and Barber [14] take advantage of the simple idea that if a term is annotated with a label in one name, it is highly likely that this term should be annotated with the same label everywhere in the data. Then they annotated this label everywhere in the corpus, assuming that the annotations are the same unless explicitly told otherwise (by further annotations). In this way, they used all the data in the corpus, containing largely only partially annotated records. To learn the parameters (the transition and emission probability tables) they use the Expectation Maximization (EM) algorithm which is an iterative algorithm that increases the likelihood of the corpus given the model parameters in each iteration.

The Yarowsky algorithm [17], originally proposed for word sense disambiguation, makes the assumption that it is very unlikely for two occurrences of a word in the same discourse to have different senses. This assumption is exploited by selecting words classified with high confidence according to sense and adding other contexts of the same words in the same discourse to the training data, even if they have low confidence. This allows the algorithm to learn new contexts for the senses leading to higher accuracy.

Wong and Hwee [16] use the idea of multiple mentions of a token sequence being to the same named entity for feature engineering. They use a named entity recognition model based on the maximum entropy framework to tag a large unlabelled corpus. Then the majority tags of the named entities are collected in lists. The model is then retrained by using these lists as extra features. This method requires a sufficient amount of manually tagged data initially to work.

Liao and Veeramachaneni [6] repeated learning to improve training corpus and the feature set by selecting unlabelled data that has been classified with low confidence by the classifier trained on the original training data, but whose labels are known with high precision from independent evidence. They propose two strategies of obtaining such independent evidence for NER. The first strategy

is based on the fact that multiple mentions of capitalized tokens are likely to have the same label and occur in independently chosen context and call that the multi-mention property. The second strategy is based on the fact that entities such as organizations, persons, etc., have context that is highly indicative of the class, yet is independent of the other context (e.g. company suffixes like Inc., Co., etc.; person titles like Mr., CEO, etc.). They use two heuristics to find the low confidence sequence tokens. In the first heuristics, if the sequence of tokens has been classified as (Organization, Person, Location) with high confidence score (larger than a threshold T), their system forces the labels of other occurrences of the same sequence in the same document, to be (Organization, Person, Location) and adds all such duplicate sequences classified with low confidence (smaller than T) to the training data for the next iteration. The second heuristics is removing company suffix or person title from the sentence. Then the system reclassifies the sentence after removing the company suffix or person title and checks whether the labels have low confidence score or not. If it has low confidence score, the sequence will be added to the training data.

Our research bases on [6] and expands it by combining proper name coreference, named-ambiguity heuristics with a powerful sequential learning model, Conditional Random Fields. This approach will be discussed in detailed in Section 4.

3 Conditional Random Field

Conditional random fields are undirected graphical models trained to maximize a conditional probability [7].

A linear-chain CRF with parameters $\lambda = \{\lambda_1, \ldots, \lambda_N\}$ defines a conditional probability for a state (or label) sequence $\mathbf{y} = y_1, \ldots, y_T$ given an input sequence $\mathbf{x} = x_1, \ldots, x_T$ to be

$$P_\lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathbf{Z_x}} \exp\left(\sum_{t=1}^{T}\sum_{k=1}^{N} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right) \quad . \tag{1}$$

where T is the length of sequence, N is the number of features, $\mathbf{Z_x}$ is the normalization constant that makes the probability of all state sequences sum to one, $f_k(y_{t-1}, y_t, \mathbf{x}, t)$ is a feature function which is often binary-valued, but can be real-valued, and λ_k is a learned weight associated with feature f_k. Large positive values for λ_k indicate a preference for such an event, while large negative values make the event unlikely.

The weights of a CRF, $\lambda = \{\lambda_1, \ldots, \lambda_N\}$, are a set to maximize the conditional log-likelihood of labelled sequences in some training set, $D = \{(\mathbf{x}^1, \mathbf{l}^1), (\mathbf{x}^2, \mathbf{l}^2), \ldots, (\mathbf{x}^M, \mathbf{l}^M)\}$:

$$L_\lambda = \sum_{j=1}^{M} \log(P_\lambda(\mathbf{l}^j|\mathbf{x}^j)) - \sum_{k=1}^{N} \frac{\lambda_k^2}{2\sigma^2} \quad . \tag{2}$$

where the second sum is a Gaussian prior over parameters (with variance σ) that provides smoothing to help cope with sparsity in the training data.

When the training labels make the state sequence unambiguous (as they often do in practice), the likelihood function in exponential models such as CRF is convex, so there are no local maxima, and thus finding the global optimum is guaranteed. It has recently been shown that quasi-Newton methods, such as L-BFGS, are significantly more efficient than traditional iterative scaling and even conjugate gradient [10].

Inference in CRF is to find the most probable state sequence $\mathbf{y}^*$ corresponding to the given observation sequence $\mathbf{x}$.

$$\mathbf{y}^* = \text{argmax}_{\mathbf{y}^*} p(\mathbf{y}|\mathbf{x}) \ . \tag{3}$$

In order to find $\mathbf{y}^*$, one can apply the dynamic programming technique with a slightly modified version of the original Viterbi algorithm for HMMs.

4 Named Entity Recognition in Vietnamese Text

The training module of our NER takes as input a small set of Vietnamese documents that have been annotated for three types of named entities including Person, Organization and Location, e.g., *Ông* <Person>*Nguyễn Cảnh Lương*</Person> *hiện giữ chức vụ phó hiệu trưởng* <Orgnization>*trường Đại học Bách khoa Hà Nội*</Orgnization > /Mr.Nguyen Canh Luong currently keeps the position of vice-president of Hanoi University of Science and Technology.

The model after one training step is used to repredict unlabelled training data. The confident score is computed to find the high confidence NEs. The low confident NEs are then found based on the confident score and heuristics.

Section 4.1 presents Vietnamese characteristics that involve the organization of named-entities, in order to create heuristics for finding low confidence NEs. From that point of view, the heuristics are used in our system are proposed. Section 4.2 introduces the semi-supervised learning algorithm for recognizing named entities.

4.1 Characteristics of Vietnamese Proper Names

There are some cases of ambiguities between entities in Vietnamese text, as shown below.

1. **Case 1:** One name or one NE is a part of another NE. More specifically, one NE may be the middle part or the last part of another NE. In this case, the NE that covers the larger text should be tagged instead of the smaller one. For example, the phrase "*Công ty Phát triển Phần mềm Tạ Quang Bửu*/The Ta Quang Buu software development company" is the name of an organization. This phrase contains the word "*Tạ Quang Bửu*" which is the name of a Vietnamese famous scientist. In this case, this phrase should be tagged as an organization name instead of tagging "*Tạ Quang Bửu*" as a person name.
2. **Case 2:** The recognition of a name entity depends on its context. For example:

Example 2.

(a) *Hôm nay,* ***công ty FPT*** *tổ chức liên hoan cho các thành viên* /Today, the FPT company organizes a party for its employees.

(b) *Hôm nay, chúng tôi sẽ tổ chức liên hoan ở* ***công ty FPT*** /Today, we organize a party at the FPT company.

The FPT company is an organization in the first sentence, whereas it is a location in the second one.

These ambiguities will be considered to create NER rules, as reported in Section 4.2.

Beside the above rules, the forms of named entities in Vietnamese are also considered to recognize NE. These forms are shown below:

- Person Names: [prefix] + [family name] + [middle name] + name
- Organization Names: [prefix] + [category] + [category of business] + name + [location]
- Location Names: [prefix] + name

In the above forms,the words in the square brackets are optional and the name can sometimes be abbreviated. This abbreviation can be placed in round brackets or not. Based on above forms, Nguyen and Cao [5] created the proper name coreference rules for Vietnamese text as shown in Table 1.

Some rules in Table 1 can be applied suitably with other languages. For example, rule 2 can be used to find name coreference in English, e.g, Steven Paul Jobs and Jobs. Specific rules for a certain language are also welcoming to make our system more efficient when being operated for that language.

4.2 Semi-supervised Learning Algorithm

Based on the idea of Liao and Veeramachaneni [6] , our system starts by training a model from a small labelled data L. This model is used to find new training data from unlabelled text. After extracting NEs by using the model getting from the training process, the low confidence NEs in unlabelled texts are detected by using heuristics for proper name coreference, some special rules, and rules for resolving ambiguity problems in labeling entities. The system is then retrained on the new data which includes low confidence NEs above. This process is repeated until the system cannot be improved. The algorithm is shown in Table 2 below.

In the training process (Step 1), training documents are first parsed by a Part of Speech (POS) tagger [8] to split documents into words and to get the syntactic roles of words in the sentences. Then features used in the CRF algorithm are calculated and a model C is build. The features are common features that are widely applied in other NER systems. These features are the word itself, the orthographic feature, the gazetteer feature and the POS of the current word, two words before and two words after the current word. In other words, all above mentioned features are calculated for the window size of five.

Table 1. Proper Name Coreferent Rules[1]

Rule	Content
1	Two names are similar.
2	One name is a part of another name, e.g., "*Nguyễn Chí Mai*" and "*Mai*".
3	One name is an alias of another name, e.g., "*Sài Gòn*" and "*TP Hồ Chí Minh*".
4	One name is an abbreviation of another name, e.g., "*TP HCM*" and "*Thành Phố Hồ Chí Minh*".
5	The first k words and the last m words of the two names are similar, in which $k + m$ is the total number of words of one name, e.g., "*Công ty Cổ phần Đại An*" and "*Công ty Đại An*".
6	Except the prefix, all words of N2 appear in N1 or are abbreviations of N1, e.g., "*Công ty TNHH Apave Việt Nam*", "*Cty Apave Việt Nam*", and "*Công ty Pave*" are names of a company.
7	One name is the postfix of another name, e.g., "*Nguyễn Chí Mai*" and "*Chí Mai*".
8	The postfix of one name is the abbreviation of words in the postfix of another word; the remaining parts of the two names are similar. For example, with two names "*Bộ Giáo dục và Đào Tạo*" and "*Bộ GD&ĐT*", the string "*GD&ĐT*" is the abbreviation of "*Giáo dục và Đào tạo*".
9	The last k words of two names are similar, the prefix of N2 is an abbreviation of the prefix of N1, in which N2 has $k + 1$ words. For example, "*Công ty HP VN*" and "*Cty HP VN*".
10	All abbreviations of N2 abbreviate for phrases in N1 and all the remaining words in N2 are appeared in N1. For example, all phrases "*Công ty TNHH Hewlett Packard Việt Nam*", "*Cty HP VN*", "*HP VN*", "*HP Việt Nam*", and "*Công ty HP Việt Nam*" are names of one company.
11	Two names appear continuously in a document in the format N1(N2), in which N2 has only one word and is recognized as a NE. For example, "*Phòng Thương mại và Công nghiệp Việt Nam (VCCI)*", "*Liên đoàn Bóng đá Việt Nam (VFF)*", and "*Tổng công ty Cao su VN (Geruco)*".

In the extracting process (Step 2), the unlabelled documents are parsed by a POS tagger and calculated all features like Step 1. Then, these documents are labelled by the classifier that is produced by using the model C received from Step 1.The documents after being labelled in this process are called labelled documents. After the labeling process, the confidence scores are calculated by using constrained forward-backward algorithm [4] (Step 2.1). This algorithm calculates the sum of probabilities of all paths passing through the constrained segment (constrained to be the assigned labels).

Then, all NEs (Person, Location, Organization) that have high confidence scores (the score is larger than a threshold t_1) will be combined with the proper name coreference rules in Table 1 to create searching patterns. The NE with a high confidence score is called a high-confident NE. Each searching pattern is accompanied by the label of the high-confident NE that produces that pattern. For examples, if the old C finds that the phrase "*Nguyễn Chi Mai*/Nguyen Chi

[1] In Table 1, N1 and N2 are two names that are being compared.

Table 2. The semi-supervised NER algorithm

Given:
- L - a small set of labelled training data
- U - unlabelled data

Loop for k iterations:
- **Step 1**: Train a model C based on L
- **Step 2**: Extract new data D based on C
  - 2.1 Classify kth portion of U and compute confidence scores
  - 2.2 Find high-confidence NE segments and use them to find proper name coreference to tag other low-confidence words
  - 2.3 Tag named entities corresponding to rules that the system cannot detect
  - 2.4 Find qualified O words
  - 2.5 Shuffle part of the NEs in the extracted data
  - 2.6 Add extracted data to D
- **Step 3**: Add D to L

Mai" is a Person entity with high confidence score, after applying proper name conference rules, the patterns "*Mai*", "*Nguyễn*/Nguyen", "*Chi Mai*" and "*Nguyễn Chi Mai*/Nguyen Chi Mai" are detected. These patterns are also tagged as Person. With these produced patterns, we perform a process of pattern matching on the labelled documents to extract matched sequences of tokens, called entity candidates. Each entity candidate is assigned a label which is similar to the label associated with the pattern it matches (Step 2.2). The candidates which have low confidence scores (the score is smaller than t_2) or have no label in terms of the old model will be added to the training data with their assigned labels (Step 2.5 and 2.6). These entity candidates are considered as "good entity candidates". This heuristics is called heuristic Group 1.

The reason for using two thresholds t_1[2] and t_2[3] is to ensure only the new knowledge that the old model does not have is added to the training data. The NEs whose confidence score is in the range of t_1 and t_2 are not used for finding new NEs and are not added to the training data to avoid potential ambiguities.

As mentioned in Section 4.1, there are several ambiguities in Vietnamese text. Due to these ambiguities, some good entity candidates found above may not be real entities, but are parts of other entities. They may have been assigned with a label different than their correct label. To solve this problem, the good entity candidates are processed further as indicated below:

1. **Post process 1:** In the labelled documents, for each of the good entity candidates, the smallest Noun Phrase (NP) containing this candidate is checked to see whether the first words of this NP is a prefix of Person, Organiza-

[2] $t\_1 = 0,95$
[3] $t\_2 = 0,85$
These thresholds are chosen based on our experimental results.

tion, or Location or not. If yes, the entity candidate will be replaced by this NP and the label of this NP is determined by the prefix mentioned above. This method allows us to find NEs with complex structures in Vietnamese documents.

Example 3.
(a) *Hôm nay, chị* ***Nguyễn Chí Mai*** *đi Sài Gòn* /Ms. Nguyen Chi Mai goes to Saigon today.
(b) *Hôm nay,* ***công ty Nguyễn Chí Mai*** *mở cửa* /The Nguyen Chi Mai company opens today.

The phrase "*Nguyễn Chí Mai*/Nguyen Chi Mai" in Example 3a is a high-confident NE that has the Person label in the old model. By applying the process of pattern matching, the system finds that "*Nguyễn Chí Mai*/Nguyen Chi Mai" in Example 3b is a good entity candidate with the Person label, although it is actually just a part of a bigger entity - "*Công ty Nguyễn Chí Mai*/The Nguyen Chi Mai Company". In this case, the system finds the smallest NP that contains the phrase "*Nguyễn Chí Mai*/Nguyen Chi Mai", which is "*Công ty Nguyễn Chí Mai*/The Nguyen Chi Mai Company". Since the first word of "*Công ty Nguyễn Chí Mai*/The Nguyen Chi Mai Company" is a prefix of Organization, this NP is used to replace our good entity candidate "*Nguyễn Chí Mai*/ Nguyen Chi Mai" in the training data with the Organization label.

2. **Post process 2:** if a good entity candidate or the NP that replaces an NE candidate in the Post process 1 is preceded by a location adverb (*ở*/in, *tại*/at, *gần*/near, etc.), this candidate or NP will be reannotated with Location.

Example 4.
(a) *Hôm nay,* ***công ty FPT*** *tổ chức liên hoan cho các thành viên* /Today, the FPT company organizes a party for its employees.
(b) *Hôm nay, chúng tôi sẽ tổ chức liên hoan ở* ***công ty FPT*** /Today, we organize a party at the FPT company.

The entity *công ty FPT*/FPT company in Example 4a is the Organization entity. It is the Location entity in Example 4b.

Besides the above heuristics, the system also uses other NER rules that are proposed by us, based on our empirical study. These rules are called Group 2 and are shown in Table 3 below (Step 2.3). Again, some modifications and/or additions that are specific for different languages can help the system adapt successfully to other languages.

In addition, to balance the training data, the word which is predicted as O by C with high confidence score and satisfies one of three conditions (contains no capitalized token, is not a number, is not in a stopword list) will be added to the training data (Step 2.4).

Since the window size is five (two words on the left and two words on the right of the current token), features of two consecutive tokens before and after the low confidence NEs are also added to the training data.

Table 3. Rules in Group 2

Rule	Definition
1	If the NP has a prefix in one of three prefix dictionaries Person, Location and Organization, it will be labelled based on the prefix.
The following rules deal with NPs that have two properties: +Having no prefix in three prefix dictionaries of Person, Location, and Organization. +Containing only one word and all letters of this word are capitalized.	
2	If a NP is followed by a chain of words conforming the following form: [**added word**][**definition word**][**numeral word**] [**word belonging to one of Person, Location, Organization dictionaries**] in which: +**Added word**: *đã*/already, *đang*/in the process of, *vẫn*/still, *rồi*/already, *sẽ*/will, etc +**Definition word**: *là*/is, *chính là*/be, *làm*/do, *chỉ*/only, etc +**Numeral word**: *các*/many, *những*/many, *mọi*/all, *một*/one, *vài*/some, etc Definition word is obligation, whereas added words and numeral words are optional. The label of this NP will be the type of the dictionary that contains it. *Example 5.* (a) ***Andrew Grove*** *là một giám đốc công ty* /Andrew Grove is a director of the company. (b) ***Hồ Chí Minh*** *là con đường huyền thoại* /Ho Chi Minh is the legendary road. In Example 5, Andrew Grove is a Person entity, whereas *Hồ Chí Minh*/Ho Chi Minh is a Location entity.
3	If a NP is preceded by a word belonging to one of two kinds: verbs which are often followed by a location (*đến*/come, *đi*/go, *tới*/reach, etc) or adverbs which often indicates a place (*tại*/at, *ở*/at, *gần*/near, etc), this NP will be labelled as the Location entity.
4	If a NP is followed by a chain of words conforming the following form: [**definition mark**][**numeral word**] [**word belonging to one of Person, Location, Organization dictionaries**] Definition marks include comma "," , figure dash "-", open parentheses "(" (these marks can usually be an indicator to a definition for its previous word in Vietnamese) Then, the NP is labelled by the type of the dictionary (Person, Location, Organization) *Example 6.* 1. ***Vinamilk****, công ty sữa lớn nhất Việt Nam, được thành lập năm 1976* /Vinamilk, the biggest dairy company in Vietnam, is established in 1976. *Vinamilk* is an Organization entity in Example 6.
5	if a NP is preceded by a chain of words conforming the following form: [**numeral word**][**word belonging to one of Person, Location, Organization Markers**][**word that supplements the meaning of the previous word**][**mark ":" or listing word**] in which: + Listing words includes: *như*/like, *gồm*/include, *gồm có*/include, etc + Supplemental word is often an adjective Then, the NP and all of the words following this NP are labelled according to the Markers (Person, Location, Organization) in the form above (the words must contain only capitalized tokens and punctuations such as comma or semi-colon are ignored) *Example 7.* 1. *Các nước tiên tiến như:* ***Mỹ, Nhật, Pháp****,... đều quan tâm đến vấn đề này* /Advanced countries like USA, Japan, France ... are all concerned about this issue. In Example 7, *Mỹ*/USA, *Nhật*/Japan, *Pháp*/France are Location entities (since the word "*nước*/country" is listed in the Location Markers).

5 Experiments and Discussion

Table 5 show the result of nine iterations of the process when the heuristics in Group 1 and Group 2 are used.

Our experiments use 900 unlabelled documents, and 50 documents labelled manually. Each document contains about 750 tokens. All of these documents are taken from newspaper websites on economics, politics, cultures and education.

50 labelled documents are taken as initial training data; and 900 unlabelled documents are used as testing data. After each round running the training process, 100 documents from those 900 unlabelled documents are taken to find low confidence NEs.

Three experiments were carried out: (i) using the two heuristics in [6]; (ii) using the heuristics in Group 1; and (iii) using the heuristics in Group 1 and Group 2. The results are shown in Table 4 below.

These experiments are evaluated based on Precision, Recall, and F-measure, in which:

- Precision (P): number of correctly assigned labels divided by the total number of labelled items.
- Recall (R): number of correctly assigned labels divided by the number of items that should have been assigned a particular label.
- F-measure: $F = \frac{2 \times P \times R}{P+R}$

Table 4 shows that when the proper name coreference heuristics are used, the results are better than when using the heuristics in [6], especially for the Location

Table 4. Experimental results

Method	Person			Location			Organization		
	P	R	F	P	R	F	P	R	F
Heuristic[6]	79.61	87.48	83.36	65.39	74.23	69.53	67.35	64.15	65.71
Group 1	86.62	90.82	88.67	78.80	85.32	81.93	72.62	80.59	76.40
Group 1 + Group 2	93.53	92.73	93.13	85.32	91.17	88.15	77.10	81.74	79.35

Table 5. Results of 9 interations when heuristics of Group 1 and Group 2 are used

Time	Person			Location			Organization		
	P	R	F	P	R	F	P	R	F
1	69.88	73.51	71.65	48.27	65.95	55.74	46.03	53.75	49.16
2	73.06	77.38	75.16	58.49	69.85	63.67	58.22	65.33	61.57
3	76.13	79.21	77.64	63.34	75.07	68.71	65.66	71.07	68.26
4	80.17	80.89	80.53	69.11	76.97	72.83	67.67	73.60	70.51
5	85.65	84.30	84.97	71.20	78.31	74.59	69.55	75.13	72.23
6	87.45	87.25	87.35	74.83	81.28	77.92	71.83	75.92	73.82
7	91.31	88.57	89.92	75.42	83.78	79.38	75.20	78.85	76.98
8	91.80	91.14	91.47	79.18	86.63	82.74	76.01	80.16	78.03
9	93.53	92.73	93.13	85.32	91.17	88.15	77.10	81.74	79.35

and Organization entities. This is because the Location and Organization entity in Vietnamese text are very complicated. When the heuristics in Group 1 and Group 2 are used, the system also receives higher F-scores.

Table 5 shows that the more new data is added to the training data, the more accurate the system is. If a bigger corpus is used, the system promises to provide a higher accuracy.

6 Conclusions

This paper presents a semi-supervised learning method for recognizing named entities in Vietnamese text. The system starts by training a model with a small labelled data set using CRFs algorithm, then the received model is used to find new training data from unlabelled text. After extracting NEs by using the model getting from the training process, the low confidence NEs in unlabelled text are detected by using heuristics for proper name coreference, some special rules, and rules for resolving ambiguity problems in labeling entities. The system is then retrained on the new data which includes these low confidence NEs. In evaluating the system, our experiments are carried out with the heuristics mentioned above and the heuristics in [6]. The experimental results show that our heuristics outperform the heuristics in [6]. Our future work includes: (i) carrying out experiments with a larger corpus; (ii) investigating other rules that can improve the accuracy of the system; and (iii) experimenting the system with other entity types.

References

1. Blum, A., Mitchell, T.: Combining Labelled and Unlabelled Data with Co-training. In: Proceedings of the Workshop on Computational Learning Theory, pp. 92–100 (1998)
2. Bikel, D., Schwartz, R., Weischedel, R.: An Algorithm that Learns What's in a Name. Machine Learning 34(1-3), 211–231 (1999)
3. Borthwick, A.: Maximum Entropy Approach to Named Entity Recognition. Ph.D. Thesis, New York University (1999)
4. Culotta, A., McCallum, A.: Confidence Estimation for Information Extraction. In: Proceeding of HLT-NAACL, pp. 109–112 (2004)
5. Nguyen, T.H., Cao, H.T.: An Approach to Entity Coreference and Ambiguity Resolution in Vietnamese Texts. Vietnamese Journal of Post and Telecommunication 19, 74–83 (2008)
6. Liao, W., Veeramachaneni, S.: A Simple Semi-supervised Algorithm for Named Entity Recognition. In: Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing, pp. 28–36 (2009)
7. Lafferty, J., McCallum, A., Pereira, F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of ICML Conference, pp. 282–290
8. Le, H.P., Roussanaly, A., Nguyen, T.M.H., Rossignol, M.: An Empirical Study of Maximum Entropy Approach for Part of Speech Tagging of Vietnamese Texts. In: Proceedings of TALN 2010 Conference, Canada (2010)

9. McCallum, A., Li, W.: Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-enhanced Lexicons. In: Proceedings of CoNLL, Canada, pp. 188–191 (2003)
10. Malouf, R.: A Comparison of Algorithms for Maximum Entropy Parameter Estimation. In: Sixth Workshop on Computational Language Learning, CoNLL (2002)
11. Mohit, B., Hwa, R.: Syntax-based semi-supervised Named Entity Tagging. In: Proceedings of the ACL Interactive Poster and Demonstration Sessions, Michigan, pp. 57–60 (2005)
12. Nguyen, C.T., Tran, T.O., Phan, X.H., Ha, Q.T.: Named Entity Recognition in Vietnamese Free-Text and Web Documents Using Conditional Random Fields. In: Proceedings of the 8th Conference on Some Selection Problems of Information Technology and Telecommunication, Hai Phong, Vietnam (2005)
13. Niu, C., Li, W., Ding, J., Rohini, K.S.: A Bootstrapping Approach to Named Entity Classification Using Successive Learner. In: Proceedings of the 41st Annual Meeting of the ACL, pp. 335–342 (2003)
14. Perrow, M., Barber, D.: Tagging of Name Record for Genealogical Data Browsing. In: Proceedings of the 6th ACM/IEEE JCDL, Chapel Hill, NC, USA, pp. 316–325 (2006)
15. Tran, Q.T., Pham, T.X.T., Ngo, Q.H., Dinh, D., Collier, N.: Named Entity Recognition in Vietnamese Using Classifier Voting. Proceedings of ACM Transactions on Asian Language Information Processing, TALIP (2007)
16. Wong, Y., Ng, H.T.: One Class per Named Entity: Exploiting Unlabelled Text for Named Entity Recognition. In: Proceedings of IJCAI, pp. 1763–1768 (2007)
17. Yarowsky, D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In: Proceedings of Meeting of the Association for Computational Linguistics, pp. 189–196 (1995)

Topic Analysis of Web User Behavior Using LDA Model on Proxy Logs

Hiroshi Fujimoto[1], Minoru Etoh[2], Akira Kinno[1], and Yoshikazu Akinaga[1]

[1] NTT DOCOMO R&D Center, 3-6, Hikarino-oka, Yokosuka-shi, Kanagawa, Japan
{fujimotoh,kinno,akinaga}@nttdocomo.co.jp
[2] Osaka University Cybermedia Center, 1-32 Machikaneyama,
Toyonaka, Osaka, Japan
etoh@ieee.org

Abstract. We propose a web user profiling and clustering framework based on LDA-based topic modeling with an analogy to document analysis in which documents and words represent users and their actions. The main technical challenge addressed here is how to symbolize web access actions, by words, that are monitored through a web proxy. We develop a hierarchical URL dictionary generated from Yahoo! Directory and a cross-hierarchical matching method that provides the function of automatic abstraction. We apply the proposed framework to 7500 students in Osaka University. The results include, for example, 24 topics such as "Technology Oriented", "Job Hunting", and "SNS-addict." The results reflect the typical interest profiles of University students, while perplexity analysis is employed to confirm the optimality of the framework.

Keywords: Web user clustering, Latent Dirichlet Allocation, topic modeling, Proxy logs based analysis.

1 Introduction

Web access user behavior analysis is, in general, the first crucial step of personalized web applications such as advertizing, recommendation, and web search. A survey by Guandong Xu et al. [10] indicated that those applications include personalization and recommendation systems [13,19,7,3], web site modification or redesign [17] and business intelligence and e-commerce [1].

To realize the analysis needed, the application system monitors web access behavior at sites, which are categorized into clients, servers and proxies. Depending on the application, the monitoring site category and modeling of user web access may differ. This paper focuses on "topic modeling" which means that documents (i.e., users) are represented as mixtures of topics (i.e., abstracted user profile components), where a topic is a probability distribution over words (i.e., user web access actions). There have been comprehensive contributions regarding the topic modeling of user web access behavior.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 525–536, 2011.

Most successful topic modeling techniques target domain-specific and application-oriented web analysis. By narrowing user actions to viewed contents, it offers excellent performance for recommendation and targeted advertisement [2,5,9,14,22,23]. The extracted topics, in other words, abstracted user intentions, enable the system to infer the user's next action. Please note that they used SVD (singular value decomposition) [8], LSI (Latent Semantic Indexing) [18] or pLSA (probabilistic Latent Semantic Analysis) [20] as the probabilistic models, since their contributions appeared in the early 2000's. As an update, LDA (Latent Dirichlet Allocation) [6] or more sophisticated models could be used instead.

The motivation for this paper lies in the authors' belief that proxy data with a better topic and action model will yield deeper user analysis, whose results are not domain-specific nor application-oriented, but rather broadened to represent social group descriptions. The research scope of this paper seems to be similar to [11], which compared LDA to pLSA for probabilistic modeling, and associated user sessions with multiple topics to describe the user sessions in terms of viewed web pages. This paper, however, focuses on the association between words (i.e., user web accesses) and the observed click streams rather than probabilistic modeling. We also use an LDA model for topic modeling though, simply taking viewed pages as words doesn't work, since a click stream contains many meaningless pages. Given a lot of proxy data, the key issue is how to select the appropriate words so as to symbolize sessions.

Our original contributions consist of

1. a word association scheme: we call it the "cross-hierarchical directory matching method". It extracts multiple words from each user session by matching against a directory database. We use Yahoo! Directory for cross-hierarchical directory matching since it resolves the ambiguity caused by multiple matches in the same domain by choosing most the abstract URL (i.e., the uppermost URL in the directory tree). Its benefits are shown in the following section.
2. an empirical study at Osaka University of proxy log analysis. The log contains 40GB click streams of 7500 students collected over several months. The study successfully visualized social groups, say 24 segments, each with clearly distinct user profiles. To be exact, the 7500 students are characterized by LDA topics such as job hunting, SNS, IT, etc.

1.1 LDA Formulation

We assume topic modeling where the user accesses Web pages under certain topics (i.e., abstracted user intentions or tasks). For example, the user accesses a certain SNS site under his latent topic "SNS-addict", or accesses a certain job site under her latent topic "Job Hunting". In this case, by applying the concepts of LDA, a Web user should correspond to a document, accessed web contents correspond to words, and their latent topics correspond to topics of documents. The observed accesses of each user are input to the LDA model, which then outputs the association between users and topics.

1.2 Cross-Hierarchical Directory Matching

Given a session with time duration t, the task is to label the session with multiple words. In the text mining domain, dictionaries and abstraction are being used with promising results [15,25]. A dictionary should cover a broad set of comprehensive concepts and words.

We use Yahoo! Directory [26] for the dictionary as it has a simple ontology structure, a category hierarchy containing paths of abstraction. For example, we extract a specific site 'The New York Times' from a session from web accesses it contains.

We may have more specific sites such as 'China - The New York Times' as a sub-category of Newspapers. In this example, we abstract those specific subcategories to the uppermost sites that appear in the session. The results mirror breadth-first search (BFS) with multiple outputs. **Our underlying assumption is that the most abstract URL that appears in the session best represents the user's intention.** Those abstracted URLs are identified from bookmarks and the landing URLs of search results. Thus, along with the directory structure, we can apply automatically adjusted abstraction to the found URLs. We call our proposal 'Cross-Hierarchical Directory matching.

2 LDA-Based Topic Modeling

User topic modeling is the action of identifying topics that web users are interested in based on their web actions. To realize it, we employ the LDA model, which was originally proposed as a probabilistic document-topic model in the document categorization domain. LDA assumes a "bag of words", i.e. each document is thought of as a vector of word counts. Each document is represented as a probability distribution of some topics, while each topic is represented as a probability distribution over a number of words. More formally, by assuming the Dirichlet distribution for per-topic word multinomial as shown in [21], the

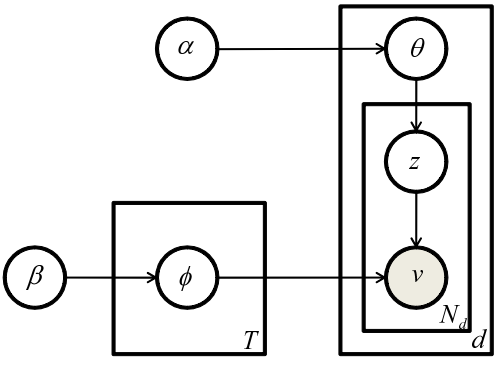


Fig. 1. Graphical model represenation of LDA

document-topic distribution $p(z|d)$ is denoted as θ and the topic-word distribution $p(v|z)$ is denoted as ϕ, where z represents a topic, d represents a document, v represents a word, α and β represent hyper-parameters , N_d is the total number of words in document d under the graphical model shown in Figure 1.

We assume topic modeling where the user accesses Web pages under certain topics (i.e., abstracted user intentions or tasks). For example, the user accesses a certain SNS site under his latent topic "SNS-addict", or accesses a certain job site under her latent topic "Job Hunting". In this case, by applying concepts of LDA, a Web user should correspond to a document, accessed web pages correspond to words, and their latent topics correspond to topics of documents. The observed accesses of each user are input to the LDA model, which then outputs the association between users and topics. In detail, under the notation shown in Table 1, the input and the outputs are as follows:

Table 1. Notations for LDA model.

D	A set of documents : D=$\{d_m \mid 1 \leq m \leq \mid D \mid \}$ where d_m represents each user.
V	A set of words : $V = \{v_w \mid 1 \leq w \leq \mid V \mid\}$ where v_w represents each accessed URLs.
Z	A set of topics : $Z = \{z_k \mid 1 \leq k \leq \mid Z \mid\}$ where z_k represents each topic.
$\nu(d_m, v_w)$	An element of matrix N.
$\theta(d_m, z_k)$	An element of matrix Θ.
$\phi(z_k, v_w)$	An element of matrix Φ.

Inputs: matrix N where each line denotes the counts of words each user accessed.

Output1: matrix Θ where each line denotes the topic distribution of each user.

Output2: matrix Φ where each line denotes word distributions of each topic.

The goal of topic modeling is to derive the optimal outputs Θ and Φ, where the topics of each user are represented by Θ and each topic is represented by Φ. To realize this, optimal input N is needed. The simplest approach, which takes all the accessed URLs as words (i.e. the approach of [11]) doesn't work, since many URLs are not related to the users' intention. Moreover, it is said in the text mining domain that word sets should be abstracted by dictionaries if a proper model is desired.

3 Symbolizing URLs from Proxy Log

Our goal is to model user-topic association. This can be realized by deriving the optimal input matrix N for the LDA model by using dictionaries in the abstraction of the original web accesses. In this section, we will show an approach based on the use of proxy logs.

3.1 Description of Proxy log

To reach our goals, we require that the proxy log for each user d_m satisfies the following conditions; each record has, at least, access time and accessed URL. The records are sorted in chronological order. A user session is also defined as a series of continuous records for each user. Each session has a time out interval δ, so the session ends when the user does not access any web page in interval δ. Formally, under the notation shown in Table 2, for each user d_m, each session $S_i^{(m)}$ consists of a series of records and each record $s_{ij}^{(m)}$ consists of access time $t_{ij}^{(m)}$ and accessed URL $l_{ij}^{(m)}$.

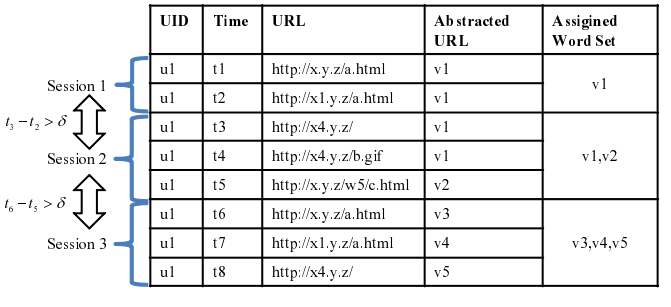
3.2 Basic Idea of Labeling Words to User Session

We define word set $V_i^{(m)}$ is the abstraction of URLs from $L_i^{(m)}$, a series of URLs in session $S_i^{(m)}$. For example, when user d_m accesses a certain SNS community site, the abstracted URL is v_w, so word v_w is assigned as the session label. Details of the abstraction process are explained in the next subsection.

An example of the relationships between sessions and words is shown in Figure 2. Each session is labeled by one or more words. For example in Session1, both URLs are abstracted to v_1 and v_1 is assigned to the session.

Table 2. Notations for Proxy log for user u_m

$S_i^{(m)}$	The $i-th$ user session : $S_i^{(m)} = \{s_{ij}^{(m)} \mid 1 \leq j \leq \mid S_i^{(m)} \mid\}$ where $s_{ij}^{(m)}$ represents $j-th$ record of the session recorded at $t_{ij}^{(m)}$.
$L_i^{(m)}$	A set of URLs accessed in session $S_i^{(m)}$: $L_i^{(m)} = \{l_{ij}^{(m)} \mid 1 \leq j \leq \mid S_i^{(m)} \mid\}$ where $l_{ij}^{(m)}$ represents accessed URL of $s_{ij}^{(m)}$.
$V_i^{(m)}$	A word set labeled to session $S_i^{(m)}$: $V_i^{(m)} = \{v_w \mid v_w \in V \cap v_w$ is labeled to session $S_i^{(m)}\}$



UID	Time	URL	Abstracted URL	Assigined Word Set
u1	t1	http://x.y.z/a.html	v1	v1
u1	t2	http://x1.y.z/a.html	v1	
u1	t3	http://x4.y.z/	v1	v1,v2
u1	t4	http://x4.y.z/b.gif	v1	
u1	t5	http://x.y.z/w5/c.html	v2	
u1	t6	http://x.y.z/a.html	v3	v3,v4,v5
u1	t7	http://x1.y.z/a.html	v4	
u1	t8	http://x4.y.z/	v5	

Fig. 2. The relationships between sessions and actions

Note that we allow multiple words to be labeled to a single session shown by Session 2 and 3 in the figure.

After all sessions of all users are labeled, a set of words V can be derived as the union of words in all sessions of all users, while the number of words accessed by each user $\nu(u_m, c_w)$ can be derived as the number of sessions labeled v_w for each user u_m. This is formally represented as follows:

$$V = \cup_{m=1}^{M} \cup_{i=1}^{|S^{(m)}|} V_i^{(m)}, N(d_m, v_w) = |\{S_i^{(m)} | \exists i : v_w \in V_i^{(m)}\}| \quad (1)$$

3.3 Cross-Hierarchical Directory matching

Cross-Hierarchical Directory matching (CHDM) [12] is a method that uses a hierarchical dictionary to get a set of abstracted URLs that are broader in concept than the originally accessed URLs. We apply CHDM to each user session to get a set of abstracted URLs in the session.

Simple examples are shown in Figure 3. The figure places a user session on the left, and the hierarchical dictionary on the right. The dictionary should have an ontology structure and a category hierarchy among URLs (To distinguish these URLs from proxy log entries, we call the former SURL.) that supports path abstraction. 6 URLs are accessed in the session, and there are 5 categories (c1-c5) and 4 SURLs are found in the dictionary.

At the matching step, URLs accessed at t1, t2, t3, t4, and t5 belong to the respective SURLs in the dictionary as in the column 'Matched SURL'. Corresponding categories of the matched SURLs are also obtained straightforwardly in the column 'Matched Category'. This yields pairs (c2, 'http://x2.y.z/'), (c3, 'http://x.z.y/'), (c4, http://x4.y.z/), and (c5, 'http://x.y.z/w/') which are assigned to the session.

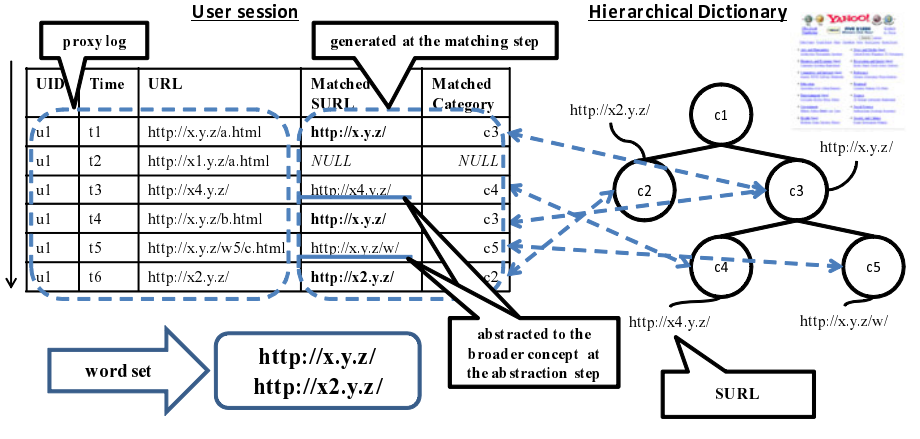


Fig. 3. Example of Cross-Hierarchical Directory matching

In the abstraction step, both 'http://x4.y.z/' and 'http://x.y.z/w/' are abstracted to 'http://x.y.z/' since corresponding categories (c4 and c5) are subordinate concepts of c3.

As a result, the set of remaining SURLS, i.e. ('http://x.y.z/', and 'http://x2.y.z') is the abstracted set of accessed web URLs in the session, and so is assigned as the word set.

4 Experiments and Results

In this section, we show two results of an experiment on a real proxy log. The first result shows the optimality of CHDM, and the second shows the 24 topics derived for students in Osaka University.

4.1 Data Sets and Evaluation Settings

We captured a set of proxy log recorded accesses from over 7500 students in Osaka University. The log, which occupied 40 GB, covered the four month period from April to July 2010. We divided the records into sessions for each user where session timeout δ was set to 1800 [sec]. This yielded 175831 sessions for 7537 users. We prepared a dictionary by crawling Yahoo! JAPAN Directory [26] in July 2010. This yielded a hierarchical dictionary with about 570 thousand distinct SURLs.

We matched the log entries against the dictionary in the manner of CHDM. This yielded, as the first result, over 20 thousand distinct words including many very minor words. We eliminated minor words (those with fewer than 4 users) to obtain 1150 test words.

We ran LDA following [24] which describes the parallel implementation of LDA and used Gibbs sampling as the inference algorithm. We set hyper-parameters α and β to $|Z|/50$ and 0.01 respectively as recommended by the authors.

4.2 Evaluation Metrics

To evaluate the optimality of the LDA model, we introduce perplexity [6], a common evaluation metric of clustering quality. Perplexity is a measure of the ability of a model to generalize documents. The better the generalization performance of a model is, the lower is the perplexity score of its output. More formally, perplexity is:

$$perplexity = \exp(-\frac{\Sigma_{m=1}^{|D|} \log p(d_{\tilde{m}})}{\Sigma_{m=1}^{|D|} N_m}) \tag{2}$$

$$\log p(d_{\tilde{m}}) = \Sigma_{w=1}^{|V|} \nu(d_{\tilde{m}}, v_w) \log p(\Sigma_{k=1}^{|Z|} \theta'_{\tilde{m}k} \phi'_{kw}) \tag{3}$$

$$\theta'_{\tilde{m}k} = \frac{\theta(d_{\tilde{m}}, z_k) + \alpha}{\Sigma_{k=1}^{|Z|}(\theta(d_{\tilde{m}}, z_k) + \alpha)}, \phi'_{kw} = \frac{\phi(z_k, v_w) + \beta}{\Sigma_{w=1}^{|V|}(\phi(z_k, v_w) + \beta)} \tag{4}$$

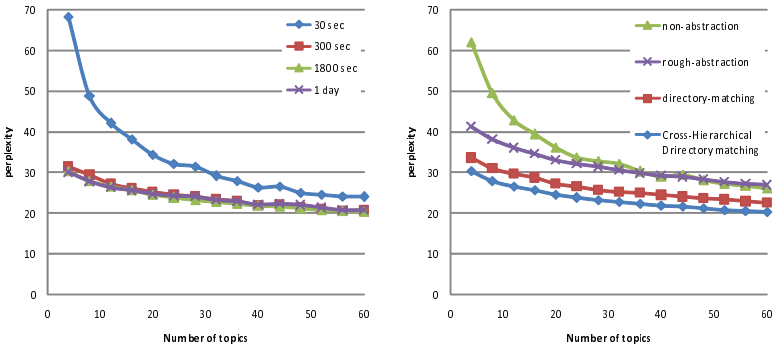


Fig. 4. Optimality Analysis of the model

where N_m is the total number of observed words for each user d_m. Perplexity is derived as cross-validated value such that $\phi(z_k, v_w)$ is derived from training set while $\theta(d_{\tilde{m}}, z_k)$ is derived from test set. (To distinguish users in the training set, users in the test set are denoted as $\tilde{m}$.) We prepared a proxy log of the data gathered over the first 3 months as the training set and the last 1 month as the test set.

4.3 Optimality Analysis of LDA Model

We first show the perplexity versus the number of topics $|Z|$ from 4 to 100; the parameter is session timeout δ with input matrix N. The plots are shown in Figure 4. Since the results show that 1800 [sec] is a good choice and yields a better LDA model than the other values, we empirically set δ to 1800 [sec].

Next we examine the optimality analysis of Cross-Hierarchical Directory matching in a comparison against other three word sets. The first is non-abstraction; that is, the word set is simply the URLs accessed in the session, i.e. $L_i^{(m)}$. This is the same approach as [11]. The second is directory-matching; the word set is the set of matched SURLs in the dictionary, i.e. $L_i'^{(m)}$. The third is rough abstraction; that is, all the URLs that contains "osaka-u" (a part of the domain name of Osaka University) are abstracted to the one word "Osaka University" after processed the Cross-Hierarchical Directory matching. This is, the abstraction does not follow the conceptual hierarchy.

The results are shown in Figure 4(right). Our method yields the lowest perplexity. In particular, its perplexity is about 10% lower than that of directory-matching (the number of topics is 24), so the abstraction ability of our approach is quite good even when the same dictionary is used. Moreover rough-abstraction suffers from the performance degradation imposed by Cross-Hierarchical Directory matching. Our clear finding is that abstraction without following the conceptual hierarchy does not work.

Table 3. 24 topics and their major words

Topic	Major Words	Topic	Major Words	Topic	Major Words
#1 MSN User	Hotmail, SkyDrive	#9 Search Books	Library of Osaka Univ.	#17 Net Shopping	Yahoo! Auctions, Amazon
#2 Video Freak	Youtube, MegaVideo	#10 Internet Equity	Yahoo! Finance	#18 Major in Engineering	Site of Engineering Osaka Univ.
#3 Full-Time Job Hunting	Recruit Portals, Job search diaries	#11 Light User	Osaka Univ. Portal	#19 Wikipedia User	Wikipedia
#4 SNS Addict	SNS sites	#12 Anonymous-Forum Addict	Anonymous Forums	#20 Part-time Job Hunting	Part-time Job Portals
#5 Making Plans to go out	Weather forecasts, Google maps	#13 Geek	Sites for Geek	#21 Writing Report	Latex learning sites
#6 Newspaper Reader	Newspaper sites	#14 News Sensitive	Yahoo! News	#22 Information Search	Question Boards
#7 Sports Fan	Yahoo! Sports	#15 Blog Watcher	Yahoo! Blog	#23 Twitterer	Twitter, Flickr
#8 Major in Bioscience	Sites about heredity or protein	#16 Geek Video Freak	Video sites for Japanese Geek	#24 Technology-Oriented	C-language learning sites

Please note that our approach is based on the heuristic assumption that the most abstracted URLs that appear in the session represents the users' intensions. Although the assumption is proved to be correct by the empirical evaluation conducted on our data sets, there is no assurance the same will be achieved with other data or other dictionaries.

4.4 Visualizing 24 Topics and Student Characterization

Finally we describe below the 24 interesting topics output by LDA. (We chose the number of topics as 24 such that all the topics involve more than 1 % students.) All the topics (named by authors) and their major words (or their description) are shown in Table 3. Each topic has distinctive words and they imply the interests or tasks of the corresponding users.

Another interesting finding is that some topics are strongly biased by the students' attributes such as grades or majors. To visualize this, we defined the pair of attribute values "science degree (x_m)" and "higher grades degree (y_m)" as implicit attributes of each user derived from the latent topics. We then modeled the associations between the latent topics and the implicit attributes for each user as a regression formula as follows:

Input:$\{\theta(d_m, z_k), a_k\}_{k=1}^{24}$

Output:a_m, where a_k is the pair of attribute values (x_k,y_k) of each topic, and a_m is the pair of implicit attribute values (x_m,y_m) of user d_m

The attribute value of each topic a_k was derived as follows. We can know which topics each student focused on by choosing the topic with maximum probability on matrix Θ. We also prepared two real attribute values, i.e. "major" and "grade" for each user. The major was taken from the students' major (science major set 1 and non-science major set -1), while the grade was taken from their grade (1st grade set 1,..., 4th grade set 4). Finally we determined a_k as follows where x_k is the average "major" and y_k is the average "grade" among the students focusing on the topic. We placed a group of students into a learning set for the formula. In the learning phase, output a_m was set to $a_{\tilde{k}}$ where $\tilde{k}$ was the topic of interest of each user. We trained the formula by Relevance Vector Regression using RVM [16], which yielded a pair of implicit attributes a_m for each student. The results are shown in Figure 5. The implicit attribute values of all 7537 users are plotted where the x-axis represents "science degree" and the y-axis represents "higher grade degree". Each point is color-coded by the user's topic of interest. The figure also plots the distribution of the number of students interested in each topic at the lower left of the figure where each topic number (from 1 to 24) mirrors the number in Table 3.

The figure shows that points with the same topic tend to cluster together. This indicates the fact that there is a strong relationship between the topics of interest and the attributes of students. Of particular interest topics points that

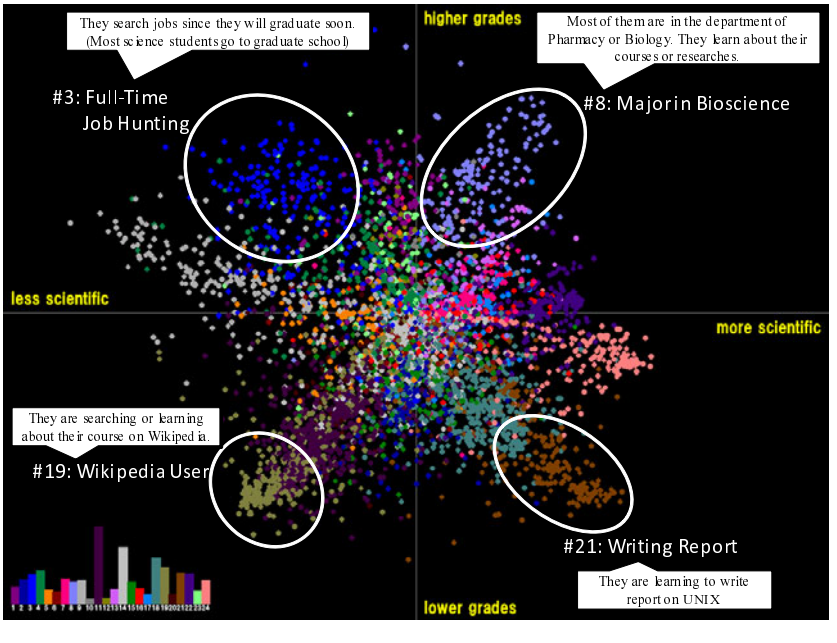


Fig. 5. Plot of implicit attribute values for each user and their latent topics

are strongly attribute-biased tend to form clearly distinct vectors. Examples of attribute-biased topics are "Full-Time Job Hunting" (#3), "Major in Bioscience" (#8), "Wikipedia User"(#19) or "Writing Report" (#21). We investigated the corresponding Web accesses in the proxy log and the summarization is as shown in the figure. On the other hand, "SNS Addict"(#4) or "Twitterer"(#23) are not biased, i.e. students use these community sites regardless of their attributes.

5 Conclusion

Profiling Web users by their interests is a key technique for many Web applications such as recommendation, site optimization, and collaborative filtering. In this paper, we proposed a user profiling method that uses the LDA model to assess Web access patterns and their latent topics. To derive an optimal model, our method employs a hierarchical URL dictionary to abstract Web accesses into broader concept words. Experiments on real proxy log data showed the optimality of our method, and also visualized 24 interesting topics. In future, we intend to apply our model to a recommendation of Web contents and evaluate the effectiveness on real application.

References

1. Buchner, A.G., Mulvenna, M.D.: Discovering Internet Marketing Intelligence through Online Analytical Web Usage Mining. The ACM SIGMOD Record 27(4), 54–61 (1998)
2. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google News Personalization: Scalable Online@Collaborative Filtering. In: Proc. of the 16th International Conference on World Wide Web, Alberta, Canada (2007)
3. Mobasher, B., Cooley, R., Srivastava, J.: Creating Adaptive Web Sites through Usage-based Clustering of URLs. In: Proc. of the 1999 Workshop on Knowledge and Data Engineering Exchange (1999)
4. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. Data Mining and Knowledge Discovery 6(1), 61–82
5. Lin, C., Xue, G.-R., Zeng, H.-J., Yu, Y.: Using probabilistic latent semantic analysis for personalized web search. In: Zhang, Y., Tanaka, K., Yu, J.X., Wang, S., Li, M. (eds.) APWeb 2005. LNCS, vol. 3399, pp. 707–717. Springer, Heidelberg (2005)
6. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research archive 3, 993–1022 (2003)
7. Weng Ngu, D.S., Wu, X.: Sitehelper: A Localized Agent That Helps Incremental Exploration of the World Wide Web. In: Proc. of the 6th International World Wide Web Conference, Santa Clara (1997)
8. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solutions. Numerische Mathematik 14(5), 403–420
9. Xu, G., Zhang, Y., Zhou, X.: Using probabilistic latent semantic analysis for Web page grouping. In: Proc. of the Research Issues in Data Engineering: Stream Data Mining and Applications (2005)

10. Xu, G., Zhang, Y., Zhou, X.: A Web Recommendation Technique Based on Probabilistic Latent Semantic Analysis. In: Proc. of the 6th International Conference on Web Information System Engineering, New York (2005)
11. Xu, G., Zhang, Y., Yi, X.: Modeling User Behavior for Web Recommendation Using LDA Model. In: Proc. of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technol, Melbourne (2008)
12. Fujimoto, H., Etoh, M., Kinno, A., Akinaga, Y.: Web User Profiling on Proxy Logs and its Evaluation in Personalization. In: Proc. of the 13th Asia-Pacific Web Conference (2011) (to appear)
13. Lieberman, H.: An Agent That Assists Web Browsing. In: Proc. of the 13th International Joint Conference on Artificial Intelligence, Montreal, Canada (1995)
14. Weng, J., Lim, E.P., Jiang, J., He, Q.: TwitterRank: finding topic-sensitive influential twitterers. In: Proc. of the 3rd ACM International Conference on Web Search and Data Mining (2010)
15. Bessho, K.: Text Segmentation Using Word Conceptual Vectors. The Transactions of Information Processing Society of Japan 42(11), 2650–2662 (2001)
16. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. The Journal of Machine Learning Research Archive 1, 211–244 (2001)
17. Perkowitz, M., Etzioni, O.: Adaptive Web Sites: Automatically Synthesizing Web Pages. In: Proc. of the 15th National Conference on Artificial Intelligence, Madison (1998)
18. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. The Journal of the American Society for Information Science 41(6), 391–407 (1990)
19. Joachims, T., Freitag, D., Mitchell, T.: Webwatcher: A Tour Guide or the World Wide Web. In: Proc. of the 15th International Joint Conference on Artificial Intelligence, Nagoya, Japan (1995)
20. Hofmann, T.: Probabilistic Latent Semantic Analysis. In: Proc. of the 22nd Annual ACM Conference on Research and Development in Information Retrieval, California (1999)
21. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. of the National Academy of Sciences of the United States of America 101, 5228–5235 (2004)
22. Jin, X., Zhou, Y., Mobasher, B.: Web usage mining based on probabilistic latent semantic analysis. In: Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle (2004)
23. Wu, X., Yan, J., Liu, N., Yan, S., Chen, Y., et al.: Probabilistic latent semantic user segmentation for behavioral targeted advertising. In: the Proc of the 3rd International Workshop on Data Mining and Audience Intelligence for Advertising, Paris (2009)
24. Wang, Y., Bai, H., Stanton, M., Chen, W.-Y., Chang, E.Y.: PLDA: Parallel Latent Dirichlet Allocation for Large-Scale Applications. In: Goldberg, A.V., Zhou, Y. (eds.) AAIM 2009. LNCS, vol. 5564, pp. 301–314. Springer, Heidelberg (2009)
25. Elberrichi, Z., Rahmoun, A., Bentaalah, M.A.: Using WordNet for Text Categorization. The International Arab Journal of Information Technology 5(1) (January 2008)
26. Yahoo! Directory, http://dir.yahoo.com/, http://dir.yahoo.co.jp/

SizeSpotSigs: An Effective Deduplicate Algorithm Considering the Size of Page Content

Xianling Mao, Xiaobing Liu, Nan Di, Xiaoming Li, and Hongfei Yan

Department of Computer Science and Technology, Peking University
{mxl,lxb,dn,lxm,yhf}@net.pku.edu.cn

Abstract. Detecting if two Web pages are near replicas, in terms of their contents rather than files, is of great importance in many web information based applications. As a result, many deduplicating algorithms have been proposed. Nevertheless, analysis and experiments show that existing algorithms usually don't work well for short Web pages[1], due to relatively large portion of noisy information, such as ads and templates for websites, existing in the corresponding files. In this paper, we analyze the critical issues in deduplicating short Web pages and present an algorithm (AF_SpotSigs) that incorporates them, which could work 15% better than the state-of-the-art method. Then we propose an algorithm (SizeSpotSigs), taking the size of page contents into account, which could handle both short and long Web pages. The contributions of SizeSpotSigs are three-fold: 1) Provide an analysis about the relation between noise-content ratio and similarity, and propose two rules of making the methods work better; 2) Based on the analysis, for Chinese, we propose 3 new features to improve the effectiveness for short Web pages; 3) We present an algorithm named SizeSpotSigs for near duplicate detection considering the size of the core content in Web page. Experiments confirm that SizeSpotSigs works better than state-of-the-art approaches such as SpotSigs, over a demonstrative Mixer of manually assessed near-duplicate news articles, which include both short and long Web pages.

Keywords: Deduplicate, Near Duplicate Detection, AF_SpotSigs, SizeSpotSigs, Information Retrieval.

1 Introduction

Detection of duplicate or near-duplicate Web pages is an important and difficult problem for Web search engines. Lots of algorithms have been proposed in recent years [6,8,20,13,18]. Most approaches can be characterized as different types of distance or overlap measures operating on the HTML strings. State-of-the-art algorithms, such as Broder et al.'s [2] and Charikar's [3], achieve reasonable precision or recall. Especially, SpotSigs[19] could avert the process of removing noise in Web page because of its smart feature selection. Existing deduplicate

[1] In this page, Web pages are classified into long (Web) pages and short (Web) page based on their core content size.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 537–548, 2011.

algorithms don't take size of the page core content into account. Essentially, these algorithms are more suitable for processing the long Web pages because they just take surfacing features to present documents. For short documents, however, the presentation is not sufficient. Especially, when documents have noise information, like ads within the Web page, the presentation is worse. Our experiments in section 5.3 also proves that the state-of-the-art deduplicate algorithm is relatively poor for short Web pages, just 0.62(F1) against 0.92(F1) for long Web pages.

In fact, there are large amount of short Web pages which have duplicated core content on the World Wide Web. At the same time, they are also very important, for example, the central bank announces some message, such as, interest rate adjustment. Fig.1 shows a pair of same-core Web pages that only differ in the framing, advertisements, and navigational banners. Both articles exhibit almost identical core contents, reporting on the match review between Uruguay and Netherlands.

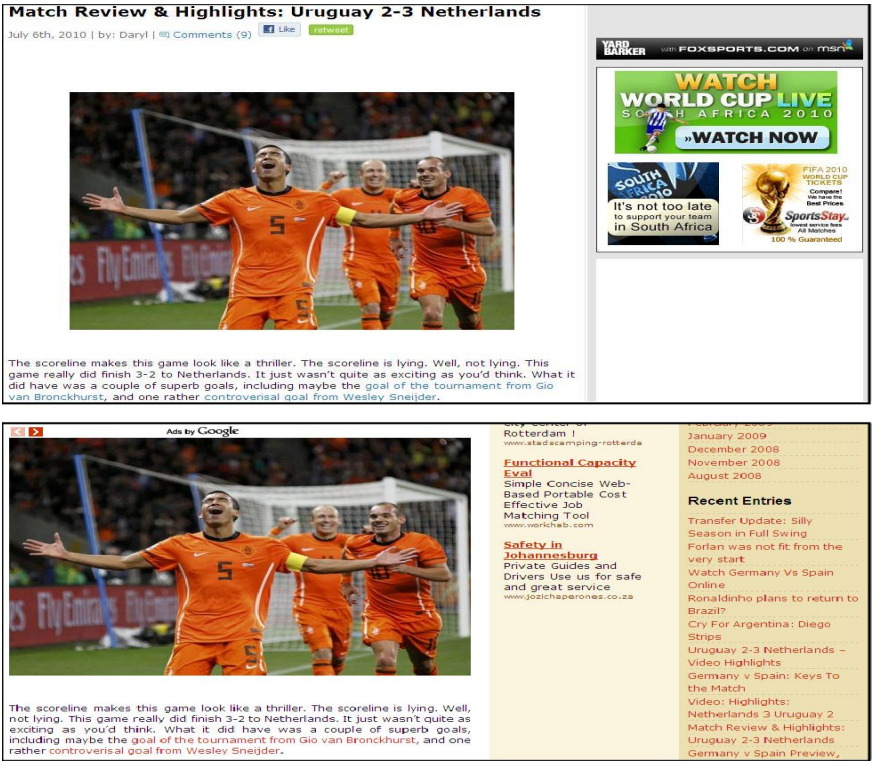


Fig. 1. Near-duplicate Web Pages: identical core content with different framing and banners(additional ads and related links removed) and the size of core contents are short

So it is important and necessary to improve the effectiveness of deduplication for short Web pages.

1.1 Contribution

1. Analyze the relation between noise-content ratio and similarity, and propose two rules of making the methods work better;
2. Based on our analysis, for Chinese, we propose 3 new features to improve the effectiveness for short Web pages, which leads to AF_SpotSigs algorithm;
3. We present an algorithm named SizeSpotSigs for near duplicate detection considering the size of the core content in Web page.

2 Related Work

There are two families of methods for near duplicate detection. One is content-based methods, the other is non-content-based methods. The content-based methods were to detect near duplicates by computing similarity between contents of documents, while the non-content-based methods made use of non-content features[10,1,17](i.e. URL pattern) to detect near duplicates. The non-content-based methods were only used to detect the near duplicate pages in one web site while the content-based methods have no any limitation. Content-based algorithms could be also divided into two groups according to whether they need noise removing. Most of the existing content-based deduplicate algorithms needed the process of removing noise.

Broder et al. [6] proposed a DSC algorithm(also called Shingling), as a method to detect near duplicates by computing similarity among the shingle sets of the documents. The similarity between two documents is computed based on the common Jaccard overlap measure between these document shingle set. In order to reduce the complexity of Shingling for processing large collections, DSC-SS(also called super shingles) was later proposed by Broder in [5]. DSC-SS makes use of meta-shingles, i.e., shingles of shingles, with only a little decrease in precision. A variety of methods for getting good shingles are investigated by Hod and Zobel [14]. Buttcher and Clarke [7] focus on Kullback-Leibler divergence in the more general context of search. A lager-scale evaluation was implemented by Henzinger[13] to compare the precision of shingling and simhash algorithms by adjusting their parameters to maintain almost same recall. The experiment shows that neither of the algorithms works well for finding near-duplicate pairs on the same site because of the influence of templates, while both achieve a higher precision for near-duplicate pairs on different sites. [21] proposed that near-duplicate clustering should incorporating information about document attributes or the content structure.

Another widespread duplicate detection technique is to generate a document fingerprint, which is a compact description of the document, and then to compute pair-wise similarity of document fingerprints. The assumption is that fingerprints can be compared much more quickly than complete documents. A common method of generating fingerprints is to select a set of character sequences from a document, and to generate a fingerprint based on the hash values of these sequences. Similarity between two documents is measured by Jaccard

formula. Different algorithms are characterized, and their computational costs are determined, by the hash functions and how character sequences are selected. Manber [18] started the research firstly. I-Match algorithm [9,16] uses external collection statistics and make recall increase by using multiple fingerprints per document. Position based schemes [4] select strings based on their offset in a document. Broder etc. [6] pick strings whose hash values are multiples of an integer. Indyk and Motwani [12,15] proposed Locality Sensitive Hashing (LSH), which is an approximate similarity search technique that scales to both large and high-dimensional data sets. There are many variant of LSH, such as LSH-Tree [3] or Hamming-LSH [11].

Generally, the noise removing is an expensive operation. If possible, the near-duplicate detection algorithm should avoid noise removing. Martin Theobald proposed SpotSigs [19] algorithm, which used word chain around stop words as features to construct feature set. For example, consider the sentence: "On a street in Milton, in the city's inner-west, one woman wept as she toured her waterlogged home." Choosing the articles a, an, the and the verb is as antecedents with a uniform spot distance of 1 and chain length of 2, we obtain the set of spot signatures S = {a:street:Milton, the:city's:inner-west}. The SpotSigs only needs a single pass over a corpus, which is much more efficient, easier to implement, and less error-prone because expensive layout analysis is omitted. Meanwhile, it remains largely independent of the input format. The method will be taken as our baseline. In this paper, considering special merits, we focus on the algorithms without noise removing, and we also take Jaccard overlap measure as our similarity measure.

3 Relation between Noise-Content Ratio and Similarity

3.1 Concepts and Notation

For calculating the similarity, we need to extract features from Web pages. We define all the features from one page as *page-feature set*; Also we split these features into *content-feature set* and *noise-feature set*. A feature comes from the core content of page is defined as *content feature (element)* and belongs to the content feature set; otherwise, the feature is called *noise feature (element)* and belongs to the noise feature set. The *noise-content (feature) ratio* represents the ratio between the size of noise feature set and the size of content feature set.

3.2 Theoretical Analysis

Let $sim(P_1, P_2) = |P_1 \cap P_2|/|P_1 \cup P_2|$ be the default Jaccard similarity as defined over two sets P_1 and P_2, each consisting of distinct page-feature set in our case. P_{1c} and P_{2c} are the content-feature sets; $P_{1n}$and P_{2n} are the noise-feature sets, which subject to $P_{1c} \cup P_{1n} = P_1$ and $P_{2c} \cup P_{2n} = P_2$. The similarity between P_{1c} and P_{2c} is $sim(P_{1c}, P_{2c}) = |P_{1c} \cap P_{2c}|/|P_{1c} \cup P_{2c}|$, which is the real value we care in the near-duplicate detection.

As we know, in fact, near-duplicate detection is to compare the similarity of the core contents of two pages, but Web pages have many noisy content, such as banners and ads. Most of algorithms is to use $sim(P_1, P_2)$ to approach $sim(P_{1c}, P_{2c})$. If $sim(P_1, P_2)$ is close to $sim(P_{1c}, P_{2c})$, it shows that the near-duplicate detection algorithm works well, and vice versa. In order to describe the difference between $sim(P_1, P_2)$ and $sim(P_{1c}, P_{2c})$, we could get the Theorem 1 as follow:

Theorem 1. Given two sets, P_1 and P_2, subject to $P_{1c} \subset P_1$, $P_{1n} \subset P_1$ and $P_{1c} \cup P_{1n} = P_1$. Similarly, $P_{2c} \subset P_2$, $P_{2n} \subset P_2$ and $P_{2c} \cup P_{2n} = P_2$; At the same time, $sim(P_1, P_2) = |P_1 \cap P_2|/|P_1 \cup P_2|$ and $sim(P_{1c}, P_{2c}) = |P_{1c} \cap P_{2c}|/|P_{1c} \cup P_{2c}|$. Let the noise-content ratio $\frac{|P_{1n}|}{|P_{1c}|} \leq \epsilon$ and $\frac{|P_{2n}|}{|P_{2c}|} \leq \epsilon$, where ϵ is a small number. Then,

$$\frac{-2\epsilon}{1+2\epsilon} \leq sim(P_1, P_2) - sim(P_{1c}, P_{2c}) \leq 2\epsilon \tag{1}$$

Proof: $let A = |P_{1c} \cap P_{2c}|, B = |P_{1c} \cup P_{2c}|$, then

$$A \leq |(P_{1c} \cup P_{1n}) \cap (P_{2c} \cup P_{2n})| \leq A + 2 * max\{|P_{1n}|, |P_{2n}|\} \tag{2}$$

$$B \leq |(P_{1c} \cup P_{1n}) \cup (P_{2c} \cup P_{2n})| \leq B + 2 * max\{|P_{1n}|, |P_{2n}|\} \tag{3}$$

From (2) and (3), we can get the following inequality:

$$\frac{A}{B + 2 * max\{|P_{1n}|, |P_{2n}|\}} \leq \frac{|(P_{1c} \cup P_{1n}) \cap (P_{2c} \cup P_{2n})|}{|(P_{1c} \cup P_{1n}) \cup (P_{2c} \cup P_{2n})|} \leq \frac{A + 2 * max\{|P_{1n}|, |P_{2n}|\}}{B} \tag{4}$$

From (4), wet get the following inequality:

$$\frac{-2A * max\{|P_{1n}|, |P_{2n}|\}}{B(B + 2 * max\{|P_{1n}|, |P_{2n}|\})} \leq \frac{|(P_{1c} \cup P_{1n}) \cap (P_{2c} \cup P_{2n})|}{|(P_{1c} \cup P_{1n}) \cup (P_{2c} \cup P_{2n})|} - \frac{A}{B} \leq \frac{2 * max\{|P_{1n}|, |P_{2n}|\}}{B} \tag{5}$$

Obviously, $A \leq B$ and $B \geq max\{|P_{1c}|, |P_{2c}|\}$. So, we get:

$$\frac{max\{|P_{1n}|, |P_{2n}|\}}{B} \leq \frac{max\{|P_{1n}|, |P_{2n}|\}}{max\{|P_{1c}|, |P_{2c}|\}} \leq \epsilon. \tag{6}$$

Another inequality is:

$$\begin{aligned}
\frac{-2A * max\{|P_{1n}|, |P_{2n}|\}}{B(B + 2 * max\{|P_{1n}|, |P_{2n}|\}} &= \frac{-2A}{B} \frac{max\{|P_{1n}|, |P_{2n}|\}}{B + 2 * max\{|P_{1n}|, |P_{2n}|\}} \\
&\geq -2 \frac{max\{|P_{1n}|, |P_{2n}|\}}{B + 2 * max\{|P_{1n}|, |P_{2n}|\}} \\
&\geq -2 \frac{max\{|P_{1n}|, |P_{2n}|\}}{max\{|P_{1c}|, |P_{2c}|\} + 2 * max\{|P_{1n}|, |P_{2n}|\}} \\
&\geq -2 \frac{\frac{max\{|P_{1n}|, |P_{2n}|\}}{max\{|P_{1c}|, |P_{2c}|\}}}{1 + \frac{2 * max\{|P_{1n}|, |P_{2n}|\}}{max\{|P_{1c}|, |P_{2c}|\}}} \\
&\geq (-2\epsilon)/(1 + 2\epsilon)
\end{aligned} \tag{7}$$

So, (5)could be reformed as:

$$-\frac{2\epsilon}{1+2\epsilon} \leq \frac{|(P_{1c} \cup P_{1n}) \cap (P_{2c} \cup P_{2n})|}{|(P_{1c} \cup P_{1n}) \cup (P_{2c} \cup P_{2n})|} - \frac{A}{B} \leq 2\epsilon \tag{8}$$

That is,

$$-\frac{2\epsilon}{1+2\epsilon} \leq sim(P_1, P_2) - sim(P_{1c}, P_{2c}) \leq 2\epsilon \tag{9}$$

□

Theorem1 shows:(1). When ϵ is small enough, the similarity $sim(P_1, P_2)$ is close to the similarity $sim(P_{1c}, P_{2c})$; (2). When ϵ reaches a certain small value, the difference between two similarity is little even though ϵ continue to become smaller, the difference varies little. That is, when noise-content ratio reaches a certain small number, the increase of effectiveness of near-duplicate detection algorithm will be little.

Without loss of generality, we assume $\frac{|P_{2n}|}{|P_{2c}|} \leq \frac{|P_{1n}|}{|P_{1c}|} = \epsilon$. Then Formula (9) could be reformed as:

$$-\frac{2|P_{1n}|}{|P_{1c}| + 2|P_{1n}|} \leq sim(P_1, P_2) - sim(P_{1c}, P_{2c}) \leq \frac{2|P_{1n}|}{|P_{1c}|} \tag{10}$$

Formula(10) shows $|P_{1c}|$ should be large for robust; Otherwise, $|P_{1c}|$ or $|P_{1n}|$ changes slightly will cause fierce change of upper bound and lower bound, which shows the algorithm is not robust. For example, assume two upper-bounds: 5/100 and 5/100, the upper bound become (5+5)/(100+100) after combining feature sets, which is equal with 5/100. but (5+1)/100 > (5+5+1)/(100+100). Obviously, (5+5)/(100+100) is more robuster than 5/100, though they have same value.

In a word, when ϵ is large relatively, we could make the algorithm work better by two rules as follows:(a). Select features that have small noise-content ratio to improve effectiveness; (b). When the noise-content ratios of two types of feature are the same, we should select the feature with larger content-feature set to make the algorithm robust, which implies that if the noise-content ratios of several types of features are very close, these features should be combined to increase the robustness while the effectiveness changes little.

4 AF_SpotSigs and SizeSpotSigs Algorithm

SpotSigs[19] provided a stopword feature, which aimed to filter natural-language text passages out of noisy Web page components, that is, noise-content ratio was small, which gave us an intuition that we should choose features that tend to occur mostly in the core content of Web documents and skip over advertisements, banners, and the navigational components. In this paper, based on thought in SpotSigs and our analysis in *section 3.2*, we developed four features which

Chinese Sign	，	。	！	？	；	：	、
English Sign	,	.	!	?	;	:	,

Chinese Stopword	的	地	得	把	了	是	与	和	每	以	都	于
Marker	De1	Di	De2	Ba	Le	Shi	Yu1	He	Mei	Yi	Dou	Yu2

Fig. 2. Table of Meaning of Chinese Punctuations and Table of Markers of Chinese Stopwords in the paper

all have small noise-content ratio. Details are as follows: **1**).*Stopword feature*; It is similar to the feature in SpotSigs that is a string of stopword and its neighboring words, except that the stopwords are different because languages are different; Because the stopwords in noisy content are less than ones in core content, so the features could decrease the noise-content ratio against Shingling features. The Chinese stopwords and corresponding marker used in this paper are listed in the Fig.2. **2**).*Chinese punctuation feature*; In English, many punctuations are the same with the special characters in HTML language. So in English, we can't use the punctuation to extract feature. In Chinese, however, this is not the case. As we known, the Chinese punctuations occurs less in the noisy area. We choose a string of punctuation and its neighboring words as Chinese punctuation feature, which makes the noise-content ratio small. The Chinese punctuations and corresponding English punctuations used in this paper are also listed in the Fig.2. **3**).*Sentence feature*; The string between two Chinese punctuations is thought as sentence; Considering the sentence with punctuation is little in noisy area, so the sentence features could decrease noise-content ratio notably. **4**).*Sentence shingling feature*; Assuming the length of one sentence is n, all 1-gram, 2-gram, ..., (n-1)-gram are taken as new features, aiming to increase the number of content-feature set for robustness and effectiveness, which would also make noise-content ratio small based on sentence feature.

The *Stopword feature* is used by the state-of-the-art algorithm, SpotSigs [19]. Though the stopwords are different because languages are different, we still call the algorithm *SpotSigs*. The experiments in Section 5.3 showed that SpotSigs could reach 0.92(F1) on long Web pages, but only 0.62 on short Web pages. Obviously, SpotSigs could not process the short Web pages well, and we need new algorithm. If all four features are used to detect near duplication, the algorithm is called *AF_SpotSigs*. The experiments in Section 5.3 showed that AF_SpotSigs could reach 0.77(F1) against 0.62(F1) of SpotSigs for short Web pages, but only increasing by 0.04(F1) with 28.8 times time overhead for long Web pages, which presents AF_SpotSigs could work better than SpotSigs for short Web pages, and the effectiveness of AF_SpotSigs is slightly better than that of SpotSigs for long Web pages but cost is higher. Considering the balance between efficiency and effectiveness, we propose algorithm called SizeSpotSigs that chooses only *stopword features* to judge the near duplication for long Web pages(namely SpotSigs) while the algorithm chooses all four-type features mentioned above for short Web pages(namely AF_SpotSigs).

5 Experiment

5.1 Data Set

For verifying our algorithms, AF_SpotSigs and SizeSpotSigs, we construct 4 datasets. Details are as follows:

Collection Shorter/Collection Longer: we construct the Collection Shorter and Longer humanly. The Collection Shorter has 379 short Web pages and 48 clusters; And the Collection Longer has 332 long Web pages and 40 clusters.

Collection Mixer/Collection Mixer_Purity: The Collection Shorter and Collection Longer are mixed as Collection Mixer, which includes 88 clusters and 711 Web pages totally. For each Web page in the Collection Mixer, we get its core content according to human judge, which lead to Collection Mixer_Purity.

5.2 Choice of Stopwords

Because quantity of stopwords is large, e.g. 370 more in Chinese, we need to select the most representative stopwords to improve performance. SpotSigs, however, just did experiments on English Collection. We don't know how to choose stopwords or the length of its neighboring words on Chinese collection. At the same time, for AF_SpotSigs, we also need to choose stopwords and the length. We find that F1 varies slightly about 1 absolute percent from a chain length of 1 to distance of 3 (figures omitted). So we choose two words as length parameter for the two algorithms.

In this section, we will seek to the best combination of stopwords for AF_Spot Sigs and SpotSigs for Chinese. We now consider variations in the choice of Spot-Sigs antecedents(stopwords and its neighboring words), thus aiming to find a good compromise between extracting characteristic signatures while avoiding an over-fitting of these signatures to particular articles or sites.

For SpotSigs, which is fit for long Web pages, the best combination was searched in the collection Longer_Sample which was sampled 1/3 clusters from the collection Longer. Moreover, for AF_SpotSigs, which is fit for short Web pages, we get the parameter over the collection Shorter_Sample, which was sampled 1/3 clusters from the collection Shorter.

Fig.3(a) shows that we obtain the best F1 result for SpotSigs from a combination of De1, Di, De2, Shi, Ba, Le, mostly occurring in core contents and less likely to occur in ads or navigational banners. Meanwhile, for AF_SpotSigs, Fig.3(b) shows the best F1 result is obtained on stopword "De1". Using a full stopword list (here we use the most frequent 40 stopwords) already tends to yield overly generic signatures but still performs good significantly.

5.3 AF_SpotSigs vs. SpotSigs

After obtaining the parameters of AF_SpotSigs and SpotSigs, we could compare the two algorithms from F1 value to computing cost. So, the two algorithms run on the Collection Shorter and Longer to do comparison.

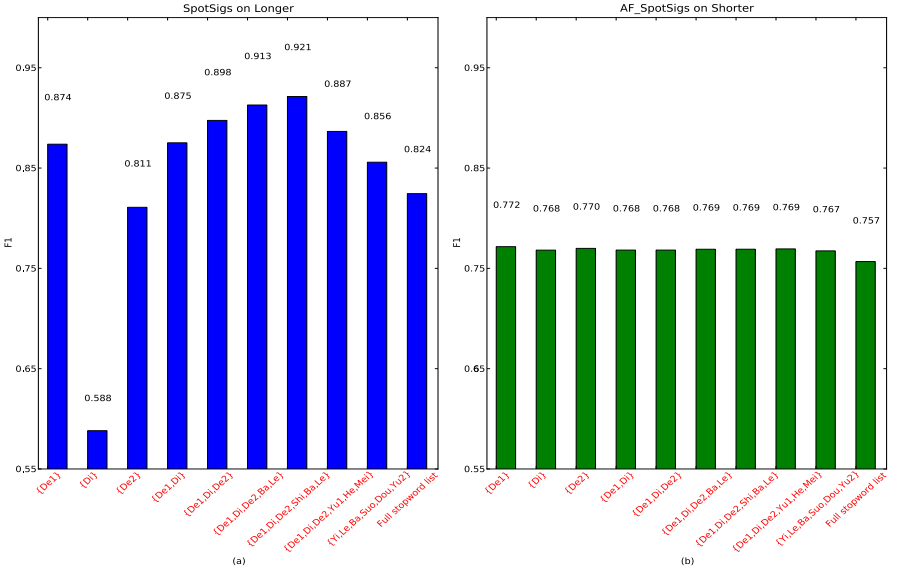


Fig. 3. (a)The effectiveness of SpotSigs with different stopwords on Longer collection;(b)The effectiveness of AF_SpotSigs with different stopwords on Shorter collection

Fig.4 shows the F1 scores of AF_SpotSigs are both better than SpotSigs on Shorter and Longer. Moreover, F1 score of SpotSigs is far worse than AF_SpotSigs on Shorter while F1 scores of two algorithms are very close on Longer. However, Table 1 shows that AF_SpotSigs took much more time than SpotSigs.

Considering balance between effectiveness and efficiency, we could partition one collection into two parts, namely the short part and long part. SpotSigs works on the long part while AF_SpotSigs runs on the short part, namely SizeSpotSigs algorithm.

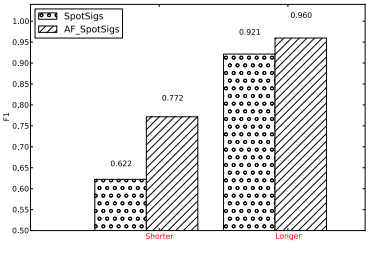


Fig. 4. The effectiveness of SpotSigs and AF_SpotSigs on Shorter and Longer

		Shorter	Longer
SpotSigs	F1	0.6223	0.9214
	Time(Sec.)	1.743	1.812
AF_SpotSigs	F1	0.7716	0.9597
	Time(Sec.)	21.17	52.31

Table 1. The F1 value and cost of two algorithms

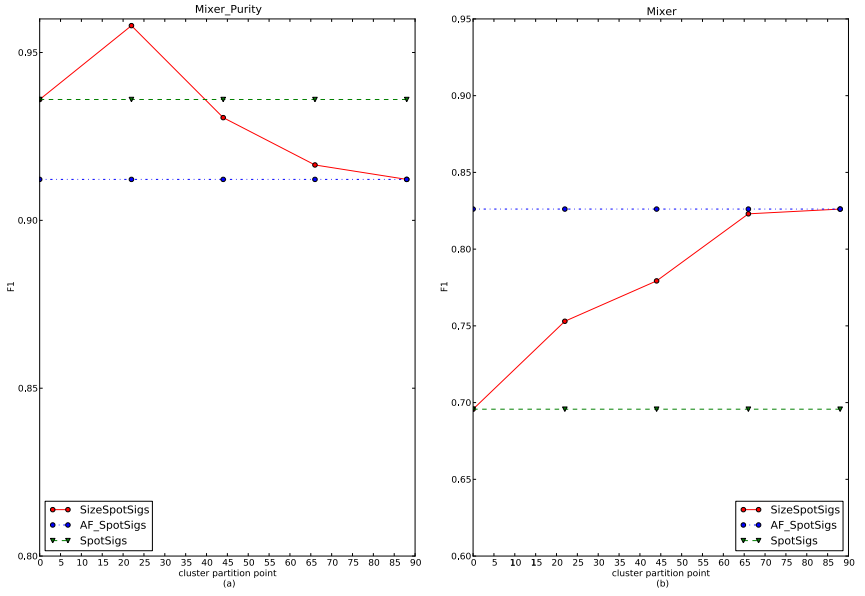


Fig. 5. F1 values of SizeSpotSigs, AF_SpotSigs and SpotSigs on Collection Mixer_purity(a) and Mixer(b)

5.4 SizeSpotSigs over SpotSigs and AF_SpotSigs

To verify SizeSpotSigs, all clusters in Mixer are sorted from small to large as their average size of core contents. We select three partition point (22,44,66) to partition set of clusters. For example, if partition point is 22, the first 22 clusters in the sorted clusters are took as small part while the rest clusters are large part. Table 2 demonstrates the nature of two parts in the every partition. Specially, 0/88 means that all clusters are took into large part which make SizeSpotSigs becomes SpotSigs while 88/0 means all clusters belong to small part which make SizeSpotSigs becomes AF_SpotSigs.

Fig.5(b) shows SizeSpotSigs works better than SpotSigs while worse than AF_SpotSigs. Moreover, the F1 value of SizeSpotSigs increases with the increase of partition point.

When purified collection is used, noise-content ratio is zero. So based on formula (9), $sim(P_1, P_2) = sim(P_{1c}, P_{2c})$, which leads to F1 value depends on $sim(P_{1c}, P_{2c})$ completely. Fig.5(a) demonstrates F1 of SizeSpotSigs rise and fall in a irregular manner, but among a reasonable interval, which all above 0.91. All details are listed in Table 3.

Table 2. The Nature of Partitions

Partition	0/88	22/66	44/44	66/22	88/0
Avg_Size(Byte)	0/2189.41	607.65/2561.43	898.24/3247.73	1290.25/4421.20	2189.41/0
File_Num	0/711	136/575	321/390	514/197	711/0

Table 3. the F1 value and time for 3 algorithms on partitions(s is Sec.)

		SpotSigs (0/88)	AF_SpotSigs (88/0)	SizeSpotSigs (22/66)	SizeSpotSigs (44/44)	SizeSpotSigs (66/22)
Mixer	F1	0.6957	0.8216	0.7530	0.7793	0.8230
	Time(s)	3.6094	148.20	7.142	22.81	61.13
Mixer_Purity	F1	0.9360	0.9122	0.9580	0.9306	0.9165
	Time(s)	2.2783	134.34	4.0118	15.99	47.00

6 Conclusions and Future Works

We analyzed the relation between noise-content ratio and similarity theoretically, which leads to two rules that could make the near-duplicate detection algorithm work better. Then, the paper proposed 3 new features to improve the effectiveness and robustness for short Web pages, which leaded to our AF_SpotSigs method.

Experiments confirm that 3 new features are effective, and our AF_SpotSigs work 15% better than the state-of-the-art method for short Web pages. Besides, SizeSpotSigs that considers the size of page core content performs better than SpotSigs over different partition points.

Future work will focus on 1). How to decide the size of the core content of Web page automatically or approximately; 2). Design more features that is fit for short Web page to improve the effectiveness, as well as generalizing the bounding approach toward other metrics such as Cosine.

Acknowledgments

This work is supported by NSFC Grant No.70903008, 60933004 and 61073082, FSSP 2010 Grant No.15. At the same time, we thank Jing He, Dongdong Shan for a quick review of our paper close to the submission deadline.

References

1. Agarwal, A., Koppula, H., Leela, K., Chitrapura, K., Garg, S., GM, P., Haty, C., Roy, A., Sasturkar, A.: URL normalization for de-duplication of web pages. In: Proceeding of the 18th ACM Conference on Information and Knowledge Management, pp. 1987–1990. ACM, New York (2009)
2. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval. Addison-Wesley, Reading (1999)
3. Bawa, M., Condie, T., Ganesan, P.: LSH forest: self-tuning indexes for similarity search. In: Proceedings of the 14th International Conference on World Wide Web, p. 660. ACM, New York (2005)
4. Brin, S., Davis, J., Garcia-Molina, H.: Copy detection mechanisms for digital documents. ACM SIGMOD Record 24(2), 409 (1995)
5. Broder, A.: Identifying and filtering near-duplicate documents. In: Giancarlo, R., Sankoff, D. (eds.) CPM 2000. LNCS, vol. 1848, pp. 1–10. Springer, Heidelberg (2000)

6. Broder, A., Glassman, S., Manasse, M., Zweig, G.: Syntactic clustering of the web. Computer Networks and ISDN Systems 29(8-13), 1157–1166 (1997)
7. Buttcher, S., Clarke, C.: A document-centric approach to static index pruning in text retrieval systems. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management, p. 189. ACM, New York (2006)
8. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, p. 388. ACM, New York (2002)
9. Chowdhury, A., Frieder, O., Grossman, D., McCabe, M.: Collection statistics for fast duplicate document detection. ACM Transactions on Information Systems (TOIS) 20(2), 191 (2002)
10. Dasgupta, A., Kumar, R., Sasturkar, A.: De-duping URLs via rewrite rules. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 186–194. ACM, New York (2008)
11. Datar, M., Gionis, A., Indyk, P., Motwani, R., Ullman, J., et al.: Finding Interesting Associations without Support Pruning. IEEE Transactions on Knowledge And Data Engineering 13(1) (2001)
12. 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, pp. 518–529. Morgan Kaufmann Publishers Inc., San Francisco (1999)
13. Henzinger, M.: Finding near-duplicate web pages: a large-scale evaluation of algorithms. In: Proceedings of the 29th annual international ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 284–291. ACM, New York (2006)
14. Hoad, T., Zobel, J.: Methods for identifying versioned and plagiarized documents. Journal of the American Society for Information Science and Technology 54(3), 203–215 (2003)
15. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, pp. 604–613. ACM, New York (1998)
16. Kołcz, A., Chowdhury, A., Alspector, J.: Improved robustness of signature-based near-replica detection via lexicon randomization. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, p. 610. ACM, New York (2004)
17. Koppula, H., Leela, K., Agarwal, A., Chitrapura, K., Garg, S., Sasturkar, A.: Learning URL patterns for webpage de-duplication. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 381–390. ACM, New York (2010)
18. Manber, U.: Finding similar files in a large file system. In: Proceedings of the USENIX Winter 1994 Technical Conference, San Fransisco, CA, USA, pp. 1–10 (1994)
19. Theobald, M., Siddharth, J., Paepcke, A.: Spotsigs: robust and efficient near duplicate detection in large web collections. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 563–570. ACM, New York (2008)
20. Whitten, A.: Scalable Document Fingerprinting. In: The USENIX Workshop on E-Commerce (1996)
21. Yang, H., Callan, J.: Near-duplicate detection by instance-level constrained clustering. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, p. 428. ACM, New York (2006)

Knowledge Transfer across Multilingual Corpora via Latent Topics

Wim De Smet[1], Jie Tang[2], and Marie-Francine Moens[1]

[1] K.U. Leuven, Leuven, Belgium
[2] Tsinghua University, Beijing, China
jie.tang@tsinghua.cn.edu,
{wim.desmet,marie-francine.moens}@cs.kuleuven.be

Abstract. This paper explores bridging the content of two different languages via latent topics. Specifically, we propose a unified probabilistic model to simultaneously model latent topics from bilingual corpora that discuss comparable content and use the topics as features in a cross-lingual, dictionary-less text categorization task. Experimental results on multilingual Wikipedia data show that the proposed topic model effectively discovers the topic information from the bilingual corpora, and the learned topics successfully transfer classification knowledge to other languages, for which no labeled training data are available.

Keywords: Cross-lingual knowledge transfer, Latent topic models, Text categorization.

1 Introduction

Cross-lingual text mining aims to transfer knowledge across different languages to help applications such as cross-lingual information retrieval, summarization and categorization in cases where translation and class-labeled resources are scarce. A specific challenge is to build models that capture content of comparable corpora, i.e. texts that to a varying degree contain shared and non-shared content. Recently, with the rapid development of online social networks, such as Facebook, MySpace, Ning, and Twitter, users have generated a huge volume of valuable multilingual language resources. However, content is seldom well-paired across the languages, forming cross-lingual parallel corpora. For example, a user may write a lot in English about her recent travel in a blog, but write only a few sentences in French. Indeed, even in well-organized Wikipedia documents, one cannot find an exact translation of a page across the different languages. However, there are many so-called "comparable" corpora available, where the documents discuss similar content in different languages, but the content in one language is not an exact translation of the content in the other language. Extracting knowledge from these comparable corpora is very valuable for cross-lingual tasks.

Current research for text categorization on bilingual corpora often focuses on learning the classification model from monolingual labeled documents and their cross-lingual pairing. Few efforts are made to directly build cross-lingual models, with which the classification knowledge can be transferred across different languages.

J.Z. Huang, L. Cao, and J. Srivastava (Eds.): PAKDD 2011, Part I, LNAI 6634, pp. 549–560, 2011.

In this paper, we address the problem of knowledge transfer across multilingual and comparable corpora. We propose a unified probabilistic model to simultaneously extract latent topics from the multilingual corpora. The learned topic models capture the common knowledge across the different languages which can be used in many applications. We apply the topic models to bilingual document classification. Our experimental results on multilingual Wikipedia data (written in English, Spanish, French and Italian) show that the proposed Bilingual Latent Dirichlet Allocation (Bi-LDA) model can effectively represent a target language without the use of any information from translation dictionaries. Additionally, the learned bilingual topic models yield a very strong categorization performance in target languages, for which no labeled training examples are available.

The rest of the paper is organized as follows: Section 2 formally formulates the problem. Section 3 explains the proposed model. Section 4 gives experimental results that validate the effectiveness of our methodology. Finally, Section 5 discusses related work and Section 6 concludes.

2 Problem Definition

In this section, we present several necessary definitions and then define the subproblems of knowledge transfer across multilingual documents addressed in this paper.

Table 1. Definition of symbols

Symbol	description
K	number of topics
$\mathcal{C}$	bilingual corpus
M	number of paired documents in $\mathcal{C}$
N_d	number of words in document d
$\mathbf{w}_d$	vector form of words in document d
θ_d	multinomial distribution over topics specific to document d
ϕ_i	multinomial distribution over words for topic z_i for the source language
ψ_i	multinomial distribution over words for topic z_i for the target language
$\mathcal{L}^{\mathcal{S}}$	labeled data in the source language S
$\mathcal{U}^{\mathcal{T}}$	unlabeled data in the target language T

A paired bilingual document corpus can be defined as $\mathcal{C} = \{(d_1^S, d_1^T), \cdots, (d_M^S, d_M^T)\}$, where (d_j^S, d_j^T) (briefly d_j for simplicity) is a pair of documents in the source language S and the target language T, respectively. Each document may describe a number of topics. For example, the Wikipedia page of a city may describe topics related to history, culture, and tourism. Before formulating the problem, we first give the definition of a probabilistic topic model.

Definition 1. ***Topic model of documents****: A topic model ϕ of a document collection D is a multinomial distribution of words $p(w|\phi_i)$ over a vocabulary V, for each ϕ_i represented in θ_d where $d \in D$. The document collection is considered a mixture of multiple topics θ.*

The underlying assumption of a topic model is that words in the document are sampled following word distributions corresponding to each topic, i.e. $p(w|\phi_i)$ and $p(\phi_i|\theta_d)$. Therefore, words with the highest probability in the distribution represent the semantic field contained in the topic. For example, the words *travel*, *tourist*, and *hotel* would represent the topic "Travel". Given this, we can define the problem of bilingual topic modeling.

Problem 1. **Bilingual topic modeling.** Given a collection of paired documents in language S and T, i.e. $\mathcal{C} = \{(d_1^S, d_1^T), \cdots, (d_M^S, d_M^T)\}$, the goal of bilingual topic modeling is to learn for every document pair a set of K topics θ, each of which defines an associated set of words in S and in T.

The topic model θ bridges knowledge across documents in two languages, which creates many potential applications. In this work, we consider a general application, i.e. bilingual document classification. In particular, we consider how to take advantage of the topic models to transfer knowledge from one language to help document classification in another language. More specifically, let $\mathcal{L}^{\mathcal{S}}$ be a labeled document collection in the source language, in which each document d_j^S is annotated with a class label $c \in \mathcal{Y}^S$, where $\mathcal{Y}^S = \{c_1, \cdots, c_{S_p}\}$ denotes the label space and S_p is the number of class labels in the source language. Let $\mathcal{U}^{\mathcal{T}}$ be a unlabeled document collection in the target language. Formally, we can define the problem of bilingual document classification as follows.

Problem 2. **Bilingual document classification.** Given a labeled document collection $\mathcal{L}^{\mathcal{S}}$ in the source language S, an unlabeled document collection $\mathcal{U}^{\mathcal{T}}$ in the target language T, and the learned bilingual topic models, the goal is to transfer the labeled supervision information from the source language to predict the label of the documents in the target language.

Please note that although we only give the definition of bilingual document modeling and classification, the problem can easily be extended to multilingual corpora.

3 Our Approach

For bilingual topic modeling, we can simply consider a general topic model as a baseline method, i.e. Latent Dirichlet Allocation (LDA) [3], to model the topic information of all bilingual documents. We propose a probabilistic topic model, referred to as Bi-LDA, to simultaneously model bilingual documents within a unified model. The model describes each pair of bilingual documents (d_j^S, d_j^T) using a common mixture θ_j, thus knowledge can be transferred across different languages via the common mixture model.

In the remainder of this section, we will first briefly review Latent Dirichlet Allocation, and then describe our proposed approach in detail.

3.1 Latent Dirichlet Allocation

Recently, probabilistic topic models attracted considerable interest and have been successfully applied to text mining tasks such as information retrieval, recommendation,

and text analysis [13,3]. Latent Dirichlet Allocation (LDA) [3] is a three-level Bayesian network, which models documents using a latent topic layer. In LDA, for each document d_j in the corpus, a multinomial distribution θ_j over topics is first sampled from a Dirichlet distribution with parameter α. Second, for each word w_{ji}, a topic z_{ji} is chosen from θ_j and the word w_{ji} is generated from a topic-specific multinomial $\phi_{z_{ji}}$. Thus, the generating probability of word w from document d_j is:

$$P(w|d_j, \theta, \phi) = \sum_{z=1}^{K} P(w|z, \phi_z) \cdot P(z|\theta_j) \tag{1}$$

In other words, it uses the topic distribution θ_j of the document and the probability ϕ_z^w of the topic generating word w to calculate the word's probability. Directly applying LDA to our bilingual problem means that the contents of both documents of a pair have to be combined together as one big document, learning a topic model were the two languages are mixed. This approach will serve as our baseline.

3.2 Bilingual Latent Dirichlet Allocation

Algorithm 3.1: BILINGUAL LDA()

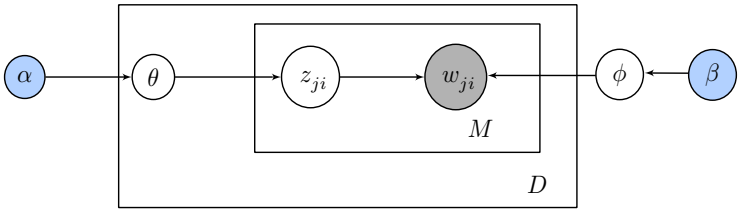
for each document pair d_j

do
- sample $\theta \sim Dir(\alpha)$
- **for each** word position $i \in d_j^S$
  - **do**
    - sample $z_{ji}^S \sim Mult(\theta_j)$
    - sample $w_{ji}^S \sim Mult(\phi, z_{ji}^S)$
- **for each** word position $i \in d_j^T$
  - **do**
    - sample $z_{ji}^T \sim Mult(\theta_j)$
    - sample $w_{ji}^T \sim Mult(\psi, z_{ji}^T)$

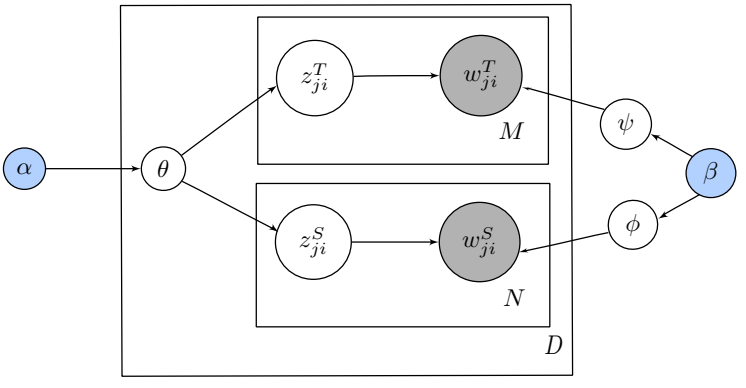
Figure 1 shows the graphical representation of the proposed model, Bilingual Latent Dirichlet Allocation (Bi-LDA). In Algorithm 3.1 we present its generative story. For each document pair d_j, a topic distribution θ is sampled from a K-dimensional Dirichlet distribution with parameter α. θ defines a distribution common to both languages (S and T). Then, each word w_{ji}^S in the source language is generated from a multinomial distribution $\phi_{z_{ji}^S}$, specific to a chosen topic z_{ji}^S. Similarly, each word w_{ji}^T of the target language is also sampled with a same procedure. We see that there is a one common θ for both languages, which implies that all topics in θ are shared across the bilingual documents.

To train this model, we used the Gibbs sampling approach. This requires two sets of formulas to converge to correct distributions: one for each topic z_{ji}^S and one for each topic z_{ji}^T. For the first, the updating formula becomes:

$$p(z_{ji}^S = k|\mathbf{w}_{d_j}^S, \mathbf{w}_{d_j}^T, \mathbf{z}_{\neg ji}^S, \mathbf{z}^T) = \frac{n_{j,k,\neg i}^S + n_{j,k}^T + \alpha}{n_{j,\cdot,\neg i}^S + n_{j,\cdot}^T + K \cdot \alpha} \cdot \frac{v_{k,w_{ji}^S,\neg}^S + \beta}{v_{k,\cdot,\neg}^S + W^S \cdot \beta} \tag{2}$$



(a) LDA



(b) BiLDA

Fig. 1. Graphical representation of LDA and Bi-LDA

where $n^S_{j,k}$ is the number of times that topic k has been sampled from the multinomial distribution specific to document d^S_j. $n^S_{j,k,\neg i}$ is the same except the current z^S_{ji} is not counted. $v^S_{k,\cdot,\neg}$ is the number of times that word w^S_{ji} in language S has been generated by topic k minus one (not including the currently associated word w^S_{ji}). In these counters a dot $(\cdot)$ means summation over all values of this variable, i.e. all topics in case of $n^S_{j,\cdot}$ and all words in $v^S_{k,\cdot,\neg}$.

For z^T_{ji}, the change in formula 2 is trivial.

3.3 Cross-Lingual Document Classification

For cross-lingual document classification, we are given a set of labeled documents in the source language and no labeled documents in the target language. The objective is to learn a classification model from the labeled documents of the source language and apply to the classification of documents in the target language. The task obviously cannot be achieved by the traditional method that only uses words as features, as there is minimal word overlap between the two languages.

Our idea for cross-lingual document classification is to take advantage of the common topics learned by the proposed topic models, to transfer the knowledge from the source language to the target language. Basically we first learn the topic models (either LDA or Bi-LDA) on a general bilingual corpus (e.g., Wikipedia). Then given a bilingual document classification task, i.e., a $\mathcal{L}^S$ in language S and an unlabeled document

collections $\mathcal{U}^T$ in language T (in the same domain as the source documents), we use the learned topic model to infer the topic distribution of each document in $\mathcal{L}^S$ and $\mathcal{U}^T$.

Each document is then taken as a data instance in the classification model and the features are defined as the inferred topic distribution. The value of each feature of an instance (e.g., document d_j^S) is the probability of the corresponding topic k in the document, i.e. $p(z = k|d_j^S)$. For the classification model, one can use any classifier such as Naive Bayes, Perceptron, Maximum Entropy, and Support Vector Machine (SVM). In this paper, we use SVM.

4 Experimental Results

4.1 Experimental Setup

Datasets: We conduct the experiments on two datasets, one (called Topic) for training and evaluating the topic models, and one (called Classification) for evaluating the transfer classification. Both datasets are downloaded from the online encyclopedia Wikipedia, from the English, Spanish, French and Italian language sections. All documents were acquired from Wikipedia *dumps*, which mirror the entire online encyclopedia, are regularly updated and can be downloaded freely.

For the Topic dataset we collect three bilingual corpora with paired documents by following "language links" between articles. Table 2 shows statistics of the Topic dataset. Note that not every article appears in each language, resulting in a different content for each bilingual dataset.

Table 2. Statistics of the Topic dataset with S = English and different values for T

	W^S	W^T	#Pair-docs
T= Spanish	29,201	27,745	18,672
T= French	27,033	20,860	18,911
T= Italian	23,346	31,388	18,898

The Classification dataset, which is different from the Topic dataset, is collected by exploiting the category-labels of Wikipedia. Specifically, we first select 5 high level categories: *books ("book")*, *films ("film")*, *programming languages ("prog")*, *sports* ("sport") and *videogames ("video")*. Then for each category, we extract up to 1,000 articles which are annotated with the category label. To acquire a workable set of documents, the categories were chosen to have examples of very broad and more specific classes. A Wikipedia article can have more than one label, and these labels can be very specific. Using the hierarchy that Wikipedia provides, we extract all subcategories for the above base classes up to three sublevels. Articles are then crawled that belonged to any one of these subcategories. Since not all Wikipedias have as large a collection of articles, we sometimes collected fewer than thousand articles for Spanish, French and Italian. Table 4.1 shows the size of the Classification dataset.

Table 3. # documents of the Classification dataset

	books	*films*	*program*	*sport*	*video*
English	1,000	1,000	1,000	1,000	1,000
Spanish	1,000	1,000	263	1,000	1,000
French	1,000	1,000	592	1,000	1,000
Italian	1,000	1,000	290	1,000	764

Comparison Methods: We compare the following methods for bilingual classification:

- *SVM+LDA*. This will be our baseline method. It combines each bilingual pair of documents into a single document, mixing the words from both languages, and employs LDA to learn the topic model from the Topic data set and then to infer the the topic distribution of the Classification data set. The two languages thus share only one common topic space. The learned topic distribution is used as the feature for SVM to learn the classification model. For SVM, we employ SVM-light.[1]
- *SVM+Bi-LDA*. This method uses the proposed Bi-LDA to learn the topic distribution from the bilingual corpora and then uses the inferred topic distribution as the features for SVM.

The code for this process is implemented in C++ and will be publicly available along with the data sets.

4.2 Perplexity

Perplexity measures a topic model's capability of predicting previously unseen documents. For a collection of these new documents (in our case the articles from the Classification dataset) C_u, it is calculated as:

$$Perp(C_u) = \exp\left(-\frac{\sum_{d \in C_u} \log\left(\prod_{w \in d} p(w)\right)}{\sum_{d \in C_u} N^d}\right) \tag{3}$$

A lower perplexity score means that the model has assigned a higher probability to the documents. Theoretically, if a model is good, i.e. has a low perplexity, it will be well adapted to the new documents and therefore yield a good representation. Although trained on paired bilingual documents, inference of each of these models has to happen on one language at a time (as we do not have any *a priori* information of the test document's content). Therefore we present the perplexity for both of the languages separately.

Table 4 lists the perplexity of each model for all three language pairs, averaged over a number of different parameter settings for the model. These settings are: (named settings will be used in further experiments)

- for LDA and Bi-LDA: ranging the number of topics from 10 to 200 in steps of 10.

[1] http://svmlight.joachims.org/

Table 4. Average perplexity of the different models, for each language pair

LDA	English - French		English - Italian		English - Spanish	
Bi-LDA	1040.3	1082.2	1139.3	1523.6	1111.9	1277.3

The difference between the perplexity of LDA and Bi-LDA can be explained easily: since the vocabulary sizes doubles by merging the two languages in the LDA model, it follows that the probability of each word halves, which then again results in a doubling of the perplexity.

4.3 Classification Accuracy

The use of knowledge transfer in a cross-lingual text categorization task is to our knowledge not studied in the literature. As a baseline we use LDA performed on concatenated texts (SVM + LDA) where the two vocabularies are mixed. Table 5 summarizes the performance of the models for each of our chosen classes, in each language pair. The F1 score is again averaged, over the same ranges of number of topics used for Table 4. It can be seen that the Bi-LDA model realizes an acceptable transfer of categorization knowledge.

Table 5. Average F1-score for the SVM classification, for each model and language pair

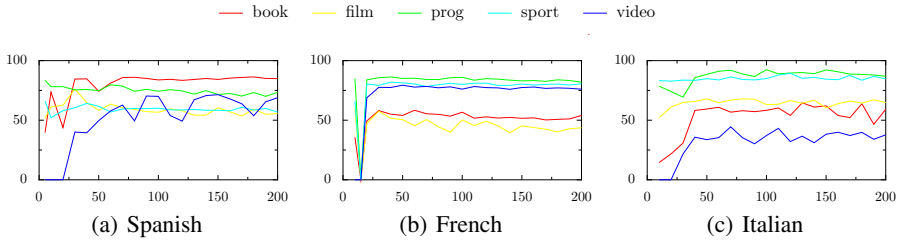
		book	*film*	*prog*	*sport*	*video*
English - Spanish	LDA	5.7	12.0	48.5	27.7	42.6
	Bi-LDA	**80.5**	**60.0**	**74.3**	**59.4**	**59.6**
English - French	LDA	3.9	1.4	67.9	42.6	**64.5**
	Bi-LDA	**53.2**	**64.0**	**86.5**	**85.4**	34.8
English - Italian	LDA	1.8	3.4	64.7	62.0	17.6
	Bi-LDA	**52.1**	**46.7**	**84.1**	**79.3**	**76.9**

In order to better assess the capabilities of the proposed topic models for cross-lingually transferring knowledge, we perform a number of additional tests.

4.4 Topic Smoothing

In this subsection, we first analyze how the bilingual document classification is influenced by different parameters (i.e., number of topics and hyperparameters), and then present a new method, called topic smoothing, to further improve the performance of bilingual document classification.

Effect of #topics on the categorization We train the Bi-LDA model with the number of topics varied, and each time we apply the learned model to the bilingual document classification. Figure 2 plots the F1-score of the classification results using Bi-LDA, when choosing a different number of topics *a priori*. We see on some categories the classification results are not very sensitive to the number of topics, except when the number is very small. But on several tasks (e.g., the classification on Italian), the results vary largely with the different number of topics.



(a) Spanish (b) French (c) Italian

Fig. 2. Bi-LDA's performance for several values of K, language combinations and categories

Topic smoothing. To deal with the above sensitivity problem, we present a strategy of topic smoothing. Basically, for Bi-LDA the smoothing method is to combine topic models that are learned with a different number of topics for bilingual document classification. We train ten topic models, with the number of topics ranging from 10 to 200 with a 20 topic step. Then we apply the topic models to inference the topic distribution of the documents in the test data set, and concatenate all the different topic distributions as features of the SVM to train the classification model. Specifically, different topic models are trained using different hyperparameters, and then topic distributions from the different topic models are concatenated as features to train the classification model. Table 6 shows the comparison of Bi-LDA and sBi-LDA. It can be seen that the smoothing method can efficiently improve (averagely +8.5%) the classification performance. As a trade-off of course, it requires substantially more computation.

Table 6. Average F1-score for the smoothed LDA compared to the average over the unsmoothed models (SBi-LDA − smoothed Bi-LDA)

		book	*film*	*prog*	*sport*	*video*
English - Spanish	Bi-LDA	80.5	60.0	74.3	59.4	59.6
	SBi-LDA	**83.6**	**60.3**	**77.7**	**61.3**	**72.5**
English - French	Bi-LDA	53.2	64.0	86.5	85.4	34.8
	SBi-LDA	**65.5**	**72.1**	**92.3**	**87.0**	**49.9**
English - Italian	Bi-LDA	52.1	46.7	84.1	79.3	76.9
	SBi-LDA	**58.9**	**53.9**	**88.1**	**83.3**	**79.1**

5 Related Work

Cross-lingual text mining is a popular research topic and quite a few research works have been conducted for cross-lingual information retrieval (e.g., [12,27,21,24]); however, cross-lingual classification is up till now rarely studied. Existing methods often rely on a translation tool to bridge documents of different languages and thus transform the problem into a monolingual classification [9,22]. Cross-lingual sentiment classification in text has recently drawn the attention [17] relying on translation dictionaries. Some efforts have also been made to reduce the number of labeled examples in both

languages using techniques such as co-training [1,25]. However, these methods still rely on a well-organized parallel multilingual corpus [11] or on a translation tool. In the work presented here we train and test on comparable corpora and do not make use of any external translation resources.

Recently, how to model the multilingual corpora so as to discover the common and differential topics among the different languages becomes an interesting research topic and many methods have been proposed. A basic idea in these methods is to use Latent Semantic Analysis (LSA) [19,6,26], document clustering [16], word alignment and machine translation [28], and parallel LDA [20,18,8] to find the correlation between different languages. While much progress has been made, two fundamental problems have been largely ignored. First, the bilingual corpora may be quite unbalanced, i.e., documents of the source language may not be comparable with the target language. The cross-lingual documents on the Web, in particular the Web 2.0, are freely authored by the users, thus the contents would be very different. Second, it is unclear how to model the bilingual documents simultaneously. Directly employing LSA, clustering, or LDA can only model the bilingual content in a common space, but cannot differentiate the topics specific to each language.

Another related work is transfer learning, which aims to transfer knowledge from a source domain to a related target domain. Two fundamental issues in transfer learning are "what to transfer" and "when to transfer". Many approaches have been proposed by reweighting instances in the source domain for use in the target domain [7]. [10] propose a locally weighted ensemble framework which can utilize different models for transferring labeled information from multiple training domains. Also many works rely on new feature representations [14,15]. For example, [2] propose a method to learn a shared low-dimensional representation for multiple related tasks and the task functions simultaneously. [23] propose to use a large amount of unlabeled data in the source domain to improve the performance on the target domain in which there are only few labeled data. They don't assume that the two domains share the class labels or topic distributions. [4] proposed a structural correspondence learning approach to induce correspondences among features from source and target domains. There are also other approaches which transfer information by shared parameters [5] or relational knowledge. Transfer learning techniques are widely used in classification, regression, clustering and dimensionality reduction problems.

The use of bilingual topic models for transferring the category knowledge across languages is completely new. Moreover, our topic models are trained on comparable corpora which are abundantly available.

6 Conclusion

In this paper we investigate knowledge transfer across multilingual corpora via latent topics. We formalize the major tasks and propose a probabilistic approach to solve the tasks. We study and compare several strategies for simultaneously modeling the content of bilingual documents. One is the bilingual topic model based on LDA. We present a sampling algorithm to learn the model. Experimental results for categorizing Wikipedia documents according to their labels demonstrate the effectiveness of the proposed transfer learning approach.

There are several directions for future work. It would be interesting to develop new algorithms to automatically find the number of topics, and detect topics that are not shared between comparable documents. As the information in different languages might be very unbalanced, the numbers of topics in the documents of different languages could also be different. Another potential issue is to apply the proposed approach to other applications (e.g., recommendation, and link prediction across multilingual documents, cross-lingual information retrieval or cross-lingual summarization) to further validate its effectiveness.

Acknowledgments

Wim De Smet is supported by WebInsight (K.U.Leuven-Tsinghua collaboration BIL/008/008) and the AMASS++ project (IWT-SBO 060051). Jie Tang is supported by NSFC(61073073, 60703059, 60973102), Chinese National Key Foundation Research (60933013, 61035004) and National High-tech R&D Program(2009AA01Z138).

References

1. Amini, M.-R., Goutte, C.: A co-classification approach to learning from multilingual corpora. Mach. Learn. 79(1-2), 105–121 (2010)
2. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Proceedings of the 18th Neural Information Processing Systems (NIPS 2006), pp. 41–48 (2006)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. JMLR 3, 993–1022 (2003)
4. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pp. 120–128 (2006)
5. Bonilla, E., Chai, K.M., Williams, C.: Multi-task gaussian process prediction. In: Proceedings of the 20th Neural Information Processing Systems (NIPS 2008), pp. 153–160 (2008)
6. Chew, P.A., Bader, B.W., Kolda, T.G., Abdelali, A.: Cross-language information retrieval using parafac2. In: KDD 2007, pp. 143–152 (2007)
7. Dai, W., Yang, Q., Xue, G.-R., Yu, Y.: Boosting for transfer learning. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), pp. 193–200 (2007)
8. De Smet, W., Moens, M.-F.: Cross-language linking of news stories on the web using interlingual topic modelling. In: CIKM-SWSM, pp. 57–64 (2009)
9. Fortuna, B., Shawe-Taylor, J.: The use of machine translation tools for cross-lingual text mining. In: ICML 2005 Workshop, KCCA (2005)
10. Gao, J., Fan, W., Jian, J., Han, J.: Knowledge transfer via multiple model local structure mapping. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008), pp. 283–291 (2008)
11. Gliozzo, A., Strapparava, C.: Exploiting comparable corpora and bilingual dictionaries for cross-language text categorization. In: ACL 2006, pp. 553–560 (2006)
12. Grefenstette, G.: Cross-Language Information Retrieval. Kluwer Academic Publishers, Norwell (1998)
13. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of Uncertainty in Artificial Intelligence, UAI, Stockholm (1999)
14. Jebara, T.: Multi-task feature and kernel selection for svms. In: Proceedings of the 21th International Conference on Machine Learning (ICML 2004) (July 2004)

15. Lee, S.-I., Chatalbashev, V., Vickrey, D., Koller, D.: Learning a meta-level prior for feature relevance from multiple related tasks. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), pp. 489–496 (July 2007)
16. Mathieu, B., Besançon, R., Fluhr, C.: Multilingual document clusters discovery. In: RIAO, pp. 116–125 (2004)
17. Mihalcea, R., Banea, C., Wiebe, J.: Learning multilingual subjective language via cross-lingual projections. In: ACL 2007 (2007)
18. Mimno, D., Wallach, H.M., Naradowsky, J., Smith, D.A., McCallum, A.: Polylingual topic models. In: EMNLP 2009, pp. 880–889 (2009)
19. Muramatsu, T., Mori, T.: Integration of plsa into probabilistic clir model. In: Proceedings of NTCIR 2004 (2004)
20. Ni, X., Sun, J.-T., Hu, J., Chen, Z.: Mining multilingual topics from wikipedia. In: WWW 2009, pp. 1155–1155 (April 2009)
21. Nie, J.-Y., Simard, M., Isabelle, P., Durand, R.: Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In: SIGIR 1999: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 74–81. ACM, New York (1999)
22. Olsson, J.S., Oard, D.W., Hajič, J.: Cross-language text classification. In: SIGIR 2005, pp. 645–646 (2005)
23. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: Transfer learning from unlabeled data. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), pp. 759–766 (June 2007)
24. Savoy, J.: Combining multiple strategies for effective monolingual and cross-language retrieval. Inf. Retr. 7(1-2), 121–148 (2004)
25. Wan, X.: Co-training for cross-lingual sentiment classification. In: ACL-IJCNLP 2009, pp. 235–243 (2009)
26. Xue, G.-R., Dai, W., Yang, Q., Yu, Y.: Topic-bridged plsa for cross-domain text classification. In: SIGIR 2008, New York, NY, USA, pp. 627–634 (2008)
27. Yang, Y., Carbonell, J.G., Brown, R.D., Frederking, R.E.: Translingual information retrieval: Learning from bilingual corpora. Artif. Intell. 103(1-2), 323–345 (1998)
28. Zhao, B., Xing, E.P.: Bitam: Bilingual topic admixture models for word alignment. In: ACL 2006 (2006)

Author Index